Decentralised and Cooperative Control of Multi-Robot Systems through Distributed Optimisation

Yi Dong[§], Zhongguo Li[†], Xingyu Zhao[§], Zhengtao Ding[‡], Xiaowei Huang[§]
[§]University of Liverpool, United Kingdom, {yi.dong, xingyu.zhao, xiaowei.huang}@liverpool.ac.uk
[†]University College London, United Kingdom, zhongguo.li@ucl.ac.uk
[‡]University of Manchester, United Kingdom, zhengtao.ding@manchester.ac.uk

ABSTRACT

Multi-robot cooperative control has gained extensive research interest due to its wide applications in civil, security, and military domains. This paper proposes a cooperative control algorithm for multi-robot systems with general linear dynamics. The algorithm is based on distributed cooperative optimisation and output regulation, and it achieves global optimum by utilising only information shared among neighbouring robots. Technically, a high-level distributed optimisation algorithm for multi-robot systems is presented, which will serve as an optimal reference generator for each individual agent. Then, based on the distributed optimisation algorithm, an output regulation method is utilised to solve the optimal coordination problem for general linear dynamic systems. The convergence of the proposed algorithm is theoretically proved. Both numerical simulations and real-time physical robot experiments are conducted to validate the effectiveness of the proposed cooperative control algorithms.

KEYWORDS

Cooperative Control, Distributed Optimisation, Optimal Coordination, Multi-robot Systems, Output Regulation

1 INTRODUCTION

Cooperation of multiple robots to accomplish complex and challenging tasks [31, 41] has been made possible by recent advances in high-performance computing, fast communication, and affordable onboard sensors. Nevertheless, it remains a challenge to design decentralised algorithms for real-world multi-robot systems. In this paper, we design an output-regulation based distributed optimisation algorithm to control the physical multi-robot systems.

On distributed multi-agent systems, several research topics, including target aggregation, trajectory tracking, containment and formation control, can be formulated as consensus problems [25]. However, when it comes to the solutions to the distributed consensus, existing methods are either designed for overly-simplified problems or based on overly-simplified models of the agents. For example, in many settings, the consensus value is defined with respect to the initial states of the agents [9, 16, 33, 40, 44], and even if this constraint was relaxed, the agents may be modelled as a single-integrator (see Equation (5) for definition) [5, 12, 42, 46]. While imposing stronger constraints leads to simpler mathematical proof on the theoretical results (such as convergence and correctness), the unrealistic constraints may result in significant gap between theoretical guarantees and practical utility.

Authors' Preprint Accepted by the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023) May 29 – June 2, 2023
ACM Conference, , AAMAS, London, U.K.. © 2023 Association for Computing Machinery.

In this paper, to address the cooperation between multiple robots, we relax the constraints from two perspectives. First, we work with consensus problems whose consensus value is defined over reference points, instead of initial states. A reference point may or may not be an initial state. Such generalisation is of practical importance, because in many applications the agents are initialised randomly and the agents' goals might be independent from their initial states. Second, we consider linear models for agents, which generalise the single-integrator model by defining the dynamics of agents with linear operators (see Equations (1b) and (1c) for definition).

While considering a more general setting, we show that the theoretical guarantees are not compromised. That is, our algorithm can achieve both correctness (Section 4.3.1) and convergence (Sections 4.3.2 and 4.3.3). This is owing to the novel distributed optimisation algorithms proposed in the paper. Distributed optimisation aims to solve an optimisation problem where the global cost function is composed of a set of N local objectives $f_i(y)$ [14], i.e., $\min_y \sum_{i=1}^{N} f_i(y)$. Due to limited communication and the requirement of local privacy protection, the local objective function $f_i(y)$ is only known to agent i. In distributed optimisation, each agent can only cooperate with its neighbours by exchanging non-sensitive information. Although this is not a new problem, existing methods [11, 26, 36] either assume single-integrator models for agents or are based on the continuous systems. For robotic systems, particularly the high-level control of the robotic systems as we aim to address in this paper, the discrete-time system and heterogeneous assumption on the robotic systems are arguably more realistic, while algorithms and their associated proofs cannot be easily transferred, which motivates our work. Intuitively, our distributed optimisation algorithm proceeds by every agent moving according to a recursive expression (i.e., considering not only on the current time but also the history) about a gradient over its local objective and the information collected from its neighbors (see Equation (8)). Also, we utilise distributed output regulation techniques to ensure that the formulated linear multi-robot systems can track the dynamic references in real-time [6, 7].

The proposed algorithm is implemented for a consensus control problem using a physical multi-robot system consisting of 4 Turtlebot robots. The communication graph of the physical robots is designed as an undirected ring. Each robot is controlled and optimised only based on the information from itself and connected neighbours. Although different robots have their respective local targets, they are eventually moved to the global optimal point since the final consensus value is generated by solving the real-time optimisation problems, which is independent of the initial states.

The major contributions of this work are summarised as follows:

(1) A distributed discrete-time cooperative control algorithm is proposed and the convergence of our algorithm has been theoretically proved. Different from most existing studies, e.g., [27, 30] and references therein, which are extensively concentrated on continuous-time systems, the proposed algorithm in this paper is ready for implementation on digital robots and platforms.
(2) A composite approach combining distributed optimisation and output regulation is developed for heterogeneous linear systems. Different from the initialisation-dependent consensus problems, the proposed approach lays the foundation for the interaction between optimisation and control.
(3) The proposed algorithm has been successfully validated on a real multi-robot system, where the errors and noises are handled by the communication among different agents. Furthermore, the reproducibility and replicability of our work are guaranteed since all source codes are available on Github for open access.

The rest of this paper is organised as follows. The mathematical preliminaries used in this paper are summarised and the researched problem is also formulated in Section 3. An output regulation based distributed optimisation approach for multi-robot consensus protocol is proposed in Section 4. Simulation results and corresponding analysis are presented in Section 5. Finally, Section 6 concludes this paper.

2 RELATED WORK

This paper studies the control algorithm of multi-agent systems. The distributed algorithm designed in this paper is based on the consensus problem that requires a distributed protocol to drive a group of agents to achieve an agreement on states.

Initiated from [22], consensus based distributed cooperative control problems have been widely studied and tactfully generalised to different specific sub-problems in recent years, such as finite-time, event-triggered, time-delayed, and switching-topology based cooperation problems. Shi and Hong considered the coordination problem of aggregation to a convex target set for a multi-agent system [33]. Zou *et al.* studied the coordinated aggregation problem of a multi-agent system while considering communication delays and applying a projection operator to guarantee the final consensus value within a target area [44]. Martinović *et al.* proposed a distributed observer-based control strategy to solve the leader-following tracking problem [16]. Wang *et al.* presented a distributed consensus and containment algorithm for the finite-time control of a multi-agent system based on time-varying feedback gain [40]. Kuriki and Namerikawa illustrated a consensus-based cooperative formation control strategy with collision-avoidance capability for a group of multiple unmanned aerial vehicles [9]. For all the above-mentioned solutions, the consensus value is hinged on the initial states of the agents, for example, the midpoint of the agents' initial locations. This is a severe restriction as in many practical scenarios, agents are initialised randomly and the goal of cooperation is without any correlation with the initial states.

To eliminate the initialisation step, the multi-agent consensus problem becomes a distributed optimisation problem when the consensus value is required to minimise the sum of local cost functions known to the individual agents. Qiu *et al.* proposed a distributed optimisation protocol to minimise the aggregate cost functions while considering both constraint and optimisation [26]. Li *et al.* designed a proportional–integral (PI) controller to solve the optimal consensus problem and introduced event-triggered communication mechanisms to reduce the communication overhead [11]. Tran *et al.* investigated two time-triggered and event-triggered distributed optimisation algorithms to reduce communication costs and energy consumption [36]. Ning *et al.* studied a fixed-time distributed optimisation protocol to guarantee the convergence within a certain steps for multi-agent system [20].

Table 1: Scope of the proposed method.

	Single Integrator		Linear System	
	Continuous-time	Discrete-time	Continuous-time	Discrete-time
Heterogeneous	-	-	[38, 45]	**this paper**
Non-heterogeneous	[9, 16, 20, 21, 26, 33, 40, 44]	[17–19]	[11, 39, 43]	[15]

In practice, it is desirable to realise consensus in discrete-time domain for real robotic applications. However, the aforementioned distributed control and optimisation works [11, 15–21, 26, 33, 36, 38–40, 43–45] can only works for either continuous or non-heterogeneous systems. As a result, it is difficult to guarantee the convergence of heterogeneous linear systems without initialisation in practical scenarios. Motivated by the observations above, in this paper, the aim of this work is to solve distributed optimal coordination problem for discrete-time and heterogeneous linear systems. We rigorously prove that consensus of heterogeneous linear systems can be achieved, while the global costs are minimised. By the interdisciplinary nature of the target problem, it spans across different problem specifics including single integrate/linear system, continuous-time/discrete-time, and heterogeneous/non-heterogeneous. We summarise and list a few related references papers to Table. 1 to highlight the scope and position of this paper.

3 PRELIMINARIES

Notations: Let $\mathbb{R}^n$ be the set of vectors with dimension $n > 0$. Let $\|x\|$ and x^T be the standard Euclidean norm and the transpose of $x \in \mathbb{R}^n$, respectively. I_p is the compatible identity matrix with dimension $p > 0$ and $\otimes$ denotes the Kronecker product. col$(\cdot)$ represents the column vector.

3.1 Graph Theory

Following [29], a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A})$ consists of $\mathcal{V} = \{v_1, \cdots, v_n\}$ as a node set and $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ as an edge set. If the node v_i is a neighbour of node v_j, then $(v_i, v_j) \in \mathcal{E}$. A directed graph is strongly connected if there exists a directed path that connects any pair of vertices. $\mathcal{A}$ is the adjacency matrix and we let $[\mathcal{A}]_{ij} = a_{ij}$ where $a_{ij} > 0$ if $(v_i, v_j) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. Let $\mathcal{D}$ be the degree matrix of graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{A})$ and $\mathcal{L} = \mathcal{D} - \mathcal{A}$ be the Laplacian matrix. We let l_{ij} be the element of matrix $\mathcal{L}$. In this paper, we assume that the information can be shared among the agents with complete information flow, formally stated in the following assumption.

Assumption 1. *The communication graph is undirected and connected.*

Under this assumption, it follows that the Laplacian matrix $\mathcal{L}$ is semi-positive definite, and zero is a simple eigenvalue of $\mathcal{L}$ with an associated eigenvector 1_N. For more detailed properties of the graph, please refer to [27].

3.2 Optimal Coordination

This paper considers an optimal coordination problem where a group of network-connected robots are designed to solve the following optimisation problem

$$\min_{y\in\mathbb{R}^p} \sum_{i=1}^{N} f_i(y) \tag{1a}$$

$$\text{s.t.}\ \ x_i(k+1) = A_i x_i(k) + B_i u_i(k) \tag{1b}$$

$$y_i(k) = C_i x_i(k) \tag{1c}$$

$$\text{for}\ \ i = 1, \cdots, N$$

where N is the number of robots and $x_i(k) \in \mathbb{R}^n$ is the state variable of the robot i that generally represents its position, speed, force and other information. $y_i(k) \in \mathbb{R}^q$ is the system output of ith robot, which can be viewed intuitively as an observation of the robotic system since the actual state of the robot is invisible in some real robotic systems. $u_i(k) \in \mathbb{R}^p$ is the system control input for robot i. For example, the control input can be the torque of a motor or the force of a robotic system. The equations (1b) and (1c) represent the discrete-time linear systems, which satisfy both additivity and homogeneity [4]. $A_i \in \mathbb{R}^{n\times n}, B_i \in \mathbb{R}^{n\times p}, C_i \in \mathbb{R}^{q\times n}$ are constant matrices, which indicate the *heterogeneous* robotic system dynamics. The linear system can be reduced to a single-integrator system if $A = I_n, B = 1^T_{n\times p}$ and $C = 1^T_{q\times n}$. $f_i(y)$ are the smooth convex function and privately known to the ith robot.

Under Assumption 1, we can reformulate the optimal coordination problem. In distributed cooperative control, each individual agent will be allocated a local decision variable, denoted as y_i. To solve the coordination problem (1), it is required that the local decision variables y_i eventually reach to the same optimal value, i.e. $y_i = y_j, \forall i, j \in \mathcal{V}$. In a connected graph, it is equivalent to require $(\mathcal{L} \otimes I_q)Y = 0$ by noting that the null-space of $\mathcal{L}$ is 1_N.

Lemma 1 ([14]). *Let Assumption 1 hold. The optimisation problem* (1) *can be equivalently reformulated as*

$$\begin{aligned} \min_{y_i\in\mathbb{R}^q,\forall i\in\mathcal{V}} &\ \sum_{i=1}^{N} f_i(y_i) \\ \textit{s.t.}\ &\ \text{(1b) } \textit{and} \text{ (1c)} \\ &\ (\mathcal{L}\otimes I_q)Y = 0 \end{aligned} \tag{2}$$

where $Y = \text{col}(y_1, y_2 \ldots, y_N)$.

Solving problem (1) is equivalent to solving the problem (2) when the graph is undirected and connected. Let $Y^* = \text{col}(y_1^*, \cdots, y_N^*)$ be the optimal solution of the problem (2), which means $y_i^* = y_j^* = y^*, \forall i, j \in \mathcal{V}$ by $(\mathcal{L} \otimes I_q)Y^* = 0$. Then, we have $\sum_{i=1}^N f_i(y_i^*) = \sum_{i=1}^N f_i(y^*)$, which implies y^* is also an optimal solution to the problem (1).

3.3 Problem Formulation

In many real world applications, the objective functions in (1a) are defined according to the distance between the agent and the position of interest, for example, robotic swarm problem [34, 35], source seeking [13, 24, 32], search and rescue [1]. In this paper, we consider agent i has an estimation of the target r_i. Its local objective is to track its local belief by optimising

$$f_i(y_i) = \|y_i - r_i\|^2 \tag{3}$$

that is, agent i intends to minimise the difference between its output and its local estimated target.

Consequently, the problem is reformulated as

$$\begin{aligned} \min_{y_i\in\mathbb{R}^q,\forall i\in\mathcal{V}} &\ \sum_{i=1}^{N} f_i(y_i) = \sum_{i=1}^{N} \|y_i - r_i\|^2 \\ \text{s.t.}\ &\ \text{(1b) and (1c)} \\ &\ (\mathcal{L}\otimes I_q)Y = 0 \end{aligned} \tag{4}$$

Here, we provide a concrete example of the reference r_i: Assuming there is a pollution source, that several robots collaborate to find. The robots have limited sensor ranges (3m). The local reference r_i of robot i is the highest concentration pollution point within the 3m range. Based on our algorithm, the robots will eventually converge to the pollution source point by communicating with neighbours and updating local targets.

Remark 1. *The formulation in* (4) *is fundamentally different from the traditional consensus control in multi-agent systems. Consensus control [27, 30] aims to drive the states/outputs of all agents to a consensus value that is determined by the initial values of the agents' states. The coordination problem in this paper is to operate the robot states/outputs to the optimal solutions of their joint cost functions* (4), *which can be either static or time-varying. It is worth noting that the references $r_i, \forall i \in \mathcal{V}$, can be generated by learning/estimation techniques using sensory information from the local onboard sensors equipped on agent i. For example, the local target could be generated based on the computer vision algorithms, such as the Yolo [28] and Apriltag [23, 37]. Here, we assume a reference point r_i is given or can be detected during the operation.*

4 ALGORITHM DEVELOPMENT AND CONVERGENCE ANALYSIS

4.1 High Level Decision-Making for Single Integrator System

Currently, agents in multi-agent systems are usually devised with effective tracking algorithm to follow the instruction given by high-level decision-makers. The dynamics of agent i can be expressed as a single integrator system:

$$y_i(k+1) = y_i(k) + v_i(k) \tag{5}$$

where $y_i(k) \in \mathbb{R}^q$ denotes the output of the ith agent. The control design can then be given by letting

$$\begin{aligned} v_i(k) &= -\beta\Big[\sum_{j\in N_i} l_{ij} y_j(k) + \sum_{j\in N_i} l_{ij}\lambda_j(k) + \nabla f_i(y_i(k))\Big] \\ \lambda_i(k+1) &= \lambda_i(k) + \beta \sum_{j\in N_i} l_{ij} y_j(k) \end{aligned} \tag{6}$$

where v_i denotes the control input of the ith agent for the integrator dynamics, l_{ij} is the element of Laplacian matrix $\mathcal{L}$, λ_i is the Lagrangian multiplier maintained by agent i, and β is the optimisation gain (equivalently the learning rate in machine learning algorithms) to be designed. We emphasise that the proposed algorithm only shares the observations $y_i(k)$ and Lagrangian multiplier $\lambda_i(k)$ among different agents, whereas the gradient term $\nabla f_i(y_i(k))$ is not transferred, and therefore the objective privacy of local agent is protected.

Denote the augmented output and Lagrangian multiplier as $Y(k) = \mathrm{col}(y_1(k), y_2(k), \dots, y_N(k))$ and $\Lambda(k) = \mathrm{col}(\lambda_1(k), \lambda_2(k), \dots, \lambda_N(k))$. Then, the algorithm above can be written in a compact form as

$$\begin{aligned} Y(k+1) =& Y(k) - \beta[(\mathcal{L} \otimes I)Y(k) + (\mathcal{L} \otimes I)\Lambda(k) + \nabla F(Y(k))] \\ \Lambda(k+1) =& \Lambda(k) + \beta(\mathcal{L} \otimes I)Y(k) \end{aligned} \tag{7}$$

where $\nabla F(Y(k)) = \mathrm{col}(\nabla f_1(y_1(k)), \nabla f_2(y_2(k)), \dots, \nabla f_N(y_N(k)))$.

As we will delineate later in the convergence analysis, iterative optimisation algorithms inherently take the form of integrators [3]. To solve distributed optimisation problems for linear systems, we resort to output regulation techniques by taking algorithm (6) as an internal reference generator.

4.2 Control Algorithm for Linear Multi-Robot Systems

The above algorithm can be extended in a nontrivial way as explained below to work with linear multi-robot systems. A distributed coordination algorithm to solve the optimal coordination problem is designed as follows.

$$\begin{aligned} u_i(k) =& -K_i x_i(k) + (G_i + K_i \Psi_i)\xi_i(k) \\ \xi_i(k+1) =& \xi_i(k) - \beta[\sum_{j \in \mathcal{N}_i} l_{ij}\xi_j(k) + \sum_{j \in \mathcal{N}_i} l_{ij}\lambda_j + \nabla f_i(\xi_i(k))] \\ \lambda_i(k+1) =& \lambda_i(k) + \beta \sum_{j \in \mathcal{N}_i} l_{ij}\xi_j \end{aligned} \tag{8}$$

where $\xi_i(k), \lambda_i(k)$ are two internal auxiliary variables to generate tracking reference for the ith agent. K_i is chosen such that $A_i - B_i K_i$ is Schur stable under assumption that the dynamics (A_i, B_i) are controllable. G_i and Ψ_i are gain matrices, which can be obtained by solving the following

$$\begin{aligned} (A_i - I)\Psi_i + B_i G_i = 0 \\ C_i \Psi_i - I = 0 \end{aligned} \tag{9}$$

Intuitively, the gain matrices K_i, G_i, Ψ_i are designed to track the reference according to system dynamics A_i, B_i, C_i. For heterogeneous systems, every robot i may have different A_i, B_i, C_i. To ensure the solvability of (9), we adopt the following assumption 2, which is a well-known regulation equation in the output regulation literature [8].

Assumption 2. *The pairs $(A_i, B_i), \forall i \in \mathcal{V}$ are controllable, and*

$$\mathrm{rank}\begin{bmatrix} A_i - I & B_i \\ C_i & 0 \end{bmatrix} = n + q. \tag{10}$$

Remark 2. *The proposed algorithm is in fact a combination of gradient-descent optimisation and output regulation techniques. The internal model $\xi_i(k)$ is generated by consensus based gradient-descent optimisation with $\lambda_i(k)$ being the Lagrangian multiplier. The design of the control input u_i is motivated by the classic output regulation approach [8].*

Similarly, the closed-loop system dynamics can be compactly written as

$$\begin{aligned} U(k) =& -KX(k) + \Pi\Xi(k) \\ \Xi(k+1) =& \Xi(k) - \beta[(\mathcal{L} \otimes I)\Xi(k) + (\mathcal{L} \otimes I)\Lambda(k) + \nabla F(\Xi(k))] \\ \Lambda(k+1) =& \Lambda(k) + \beta(\mathcal{L} \otimes I)Y(k) \end{aligned} \tag{11}$$

where $K = \mathrm{diag}(K_1, \dots, K_N)$ and $\Pi = \mathrm{diag}(G_1 + K_1\Psi_1, \dots, G_N + K_N\Psi_N)$.

4.3 Convergence Analysis

The convergence analysis of the proposed algorithm proceeds in three steps. In the first step 4.3.1, we prove that the equilibrium point of the proposed algorithm is the optimal solution of the problem (4). After that, the proposed algorithm can guarantee that both single integrator and linear system will converge to the equilibrium point, which are proven in step 4.3.2 and 4.3.3, respectively.

4.3.1 . We begin with the convergence analysis for high-level decision making in Section 4.1, which will then serve as a reference generator later in the proof of linear system regulation.

In (7), the equilibrium point, denoted as (Y^*, Λ^*), satisfy

$$\begin{aligned} \beta[(\mathcal{L} \otimes I)Y^* - (\mathcal{L} \otimes I)\Lambda^* - \nabla F(Y^*)] = 0 \\ \beta(\mathcal{L} \otimes I)Y^* = 0. \end{aligned} \tag{12}$$

Invoking the properties of $\mathcal{L}$ under Assumption 1, (12) yields

$$1_N^T \nabla F(Y^*) = 0 \tag{13}$$

Since $(\mathcal{L} \otimes I)Y^* = 0$ implies $y_i = y_j = y^*, \forall i, j \in \mathcal{V}$, it follows from (13) that

$$\sum_{i=1}^{N} \nabla f_i(y^*) = 0. \tag{14}$$

Note that the uniqueness of y^* is guaranteed as the local objective functions are all strongly convex. According to the primal-dual theory [10], the solution pair (Y^*, Λ^*) is in fact a saddle point of the Lagrangian function of $\phi(Y, \Lambda) = \sum_{i=1}^{N} f_i(y_i) + \Lambda^T(\mathcal{L} \otimes I_q)Y$.

Now, we are ready to give the convergence of (7) to the equilibrium (Y^*, Λ^*) for single integrator dynamics. It is noticed that the equilibrium points of algorithm (6) and (8) are same.

4.3.2 . To give a complete proof, we next show that the proposed algorithm can make the system asymptotically converge to the equilibrium point.

Theorem 1. *Let Assumption 1 hold. If the step size β is chosen according to $0 < \beta < \min\{\frac{1}{2\lambda_{max}(\mathcal{L})}, \frac{3}{2L}\}$ with $\lambda_{max}(\mathcal{L})$ being the maximum eigenvalue of the Laplacian matrix and L being the Lipschitz constant of the cost function, then algorithm (7) converges to the optimal solution (Y^*, Λ^*).*

Proof. From the second update Equation in (7), we have

$$\begin{aligned} \Lambda(k) - \Lambda^* =& \Lambda(k+2) - \Lambda^* - \beta(\mathcal{L} \otimes I)(Y(k) - Y^*) \\ & - \beta(\mathcal{L} \otimes I)(Y(k+1) - Y^*) \end{aligned} \tag{15}$$

Left-multiplying both sides of (15) yields

$$\begin{aligned} (\mathcal{L} \otimes I)(\Lambda(k) - \Lambda^*) =& (\mathcal{L} \otimes I)(\Lambda(k+2) - \Lambda^*) \\ & - \beta(\mathcal{L}^2 \otimes I)(Y(k) - Y^*) \\ & - \beta(\mathcal{L}^2 \otimes I)(Y(k+1) - Y^*) \end{aligned} \tag{16}$$

where we have used the property of Kronecker product $(A\otimes B)(C\otimes D) = (AC)\otimes(BD)$. Denote $Z(k+1) = Y(k) - \beta\nabla F(Y(k)) - \beta(\mathcal{L}\otimes I)(\Lambda(k)+Y(k)) - Y(k+1)$ and $\mathcal{W} = (I-\beta\mathcal{L}+\beta^2\mathcal{L}^2)\otimes I$. Combining (16) and (7), we have:

$$\begin{aligned}
&Y(k+1) - Y^* \\
=&Y(k) - Y^* - \beta\nabla F(Y(k)) - \beta(\mathcal{L}\otimes I)(\Lambda(k) + Y(k)) - Z(k+1) \\
=&Y(k) - Y^* - \beta\nabla F(Y(k)) - \beta(\mathcal{L}\otimes I)(Y(k) - Y^*) \\
&- \beta(\mathcal{L}\otimes I)(\Lambda(k) - \Lambda^*) - \beta(\mathcal{L}\otimes I)\Lambda^* - Z(k+1) \\
=&Y(k) - Y^* - \beta(\nabla F(Y(k)) - \nabla F(Y^*)) - \beta(\mathcal{L}\otimes I)(Y(k) - Y^*) \\
&- \beta(\nabla F(Y^*) + (\mathcal{L}\otimes I)\Lambda^*) - \beta((\mathcal{L}\otimes I)(\Lambda(k+2) - \Lambda^*) \\
&- \beta((\mathcal{L}^2\otimes I)(Y(k) - Y^*) - \beta(\mathcal{L}^2\otimes I)(Y(k+1) - Y^*))) \\
&- Z(k+1) \\
=&((I - \beta\mathcal{L} + \beta^2\mathcal{L}^2)\otimes I)(Y(k) - Y^*) - Z(k+1) \\
&- \beta((\mathcal{L}\otimes I)(\Lambda(k+2) - \Lambda^*) - \beta(\nabla F(Y(k)) - \nabla F(Y^*)) \\
&- \beta(\nabla F(Y^*) + (\mathcal{L}\otimes I)\Lambda^*) + \beta^2(\mathcal{L}^2\otimes I)(Y(k+1) - Y^*)
\end{aligned} \tag{17}$$

Substracting $\mathcal{W}(Y(k+1) - Y^*)$ from both sides of Equation (17), a new recursive update law is written as

$$\begin{aligned}
&\mathcal{W}(Y(k+1) - Y(k)) + \beta(\mathcal{L}\otimes I)(\Lambda(k+2) - \Lambda^*) \\
&= - (\beta\mathcal{L}\otimes I - 2\beta^2\mathcal{L}^2\otimes I)(Y(k+1) - Y^*) - Z(k+1) \\
&\quad - \beta(\nabla F(Y(k)) - \nabla F(Y^*)) - \beta(\nabla F(Y^*) + (\mathcal{L}\otimes I)\Lambda^*).
\end{aligned} \tag{18}$$

With this new recursive algorithm, the convergence can be established using Lyapunov method and saddle point dynamics.

We formulate a Lyapunov function as follows:

$$V(Y,\Lambda) = \langle Y - Y^*, \mathcal{W}(Y - Y^*)\rangle + \langle \Lambda - \Lambda^*, \Lambda - \Lambda^*\rangle \tag{19}$$

Here, we use $\langle x, y\rangle$ to represent the inner product of vectors x and y. Then we have:

$$\begin{aligned}
&V(Y(k+1), \Lambda(k+2)) - V(Y(k), \Lambda(k+1)) \\
=&\langle Y(k+1) - Y^*, \mathcal{W}(Y(k+1) - Y^*)\rangle + \|\Lambda(k+2) - \Lambda^*\|^2 \\
&- \langle Y(k) - Y^*, \mathcal{W}(Y(k) - Y^*)\rangle + \|\Lambda(k+1) - \Lambda^*\|^2 \\
=&- \langle Y(k+1) - Y(k), \mathcal{W}(Y(k+1) - Y(k))\rangle \\
&+ 2\langle Y(k+1) - Y^*, \mathcal{W}(Y(k+1) - Y(k))\rangle \\
&- \|\Lambda(k+2) - \Lambda(k+1)\|^2 \\
&+ 2\langle \Lambda(k+2) - \Lambda(k+1), \Lambda(k+2) - Y^*\rangle
\end{aligned} \tag{20}$$

Based on the Equations (11) and (12), we can derive the last term of Equation (20) as:

$$\begin{aligned}
&\langle \Lambda(k+2) - \Lambda(k+1), \Lambda(k+2) - Y^*\rangle \\
=&\langle \beta(\mathcal{L}\otimes I)(Y(k+1) - Y^*), \Lambda(k+2) - \Lambda^*\rangle \\
=&\langle \beta(\mathcal{L}\otimes I)(\Lambda(k+2) - \Lambda^*), Y(k+1) - Y^*\rangle
\end{aligned} \tag{21}$$

Therefore, we can obtain:

$$\begin{aligned}
&\langle Y(k+1) - Y^*, \mathcal{W}(Y(k+1) - Y(k))\rangle \\
&+ \langle \Lambda(k+2) - \Lambda(k+1), \Lambda(k+2) - Y^*\rangle \\
=&\langle Y(k+1) - Y^*, \beta(\mathcal{L}\otimes I)(\Lambda(k+2) - \Lambda^*) \\
&+ \mathcal{W}(Y(k+1) - Y(k))\rangle
\end{aligned} \tag{22}$$

With Equation (18), we can further derive as:

$$\begin{aligned}
&\langle Y(k+1) - Y^*, \mathcal{W}(Y(k+1) - Y(k))\rangle \\
&+ \langle \Lambda(k+2) - \Lambda(k+1), \Lambda(k+2) - Y^*\rangle \\
=&- \langle Y(k+1) - Y^*, (\beta\mathcal{L}\otimes I - 2\beta^2\mathcal{L}^2\otimes I)(Y(k+1) - Y^*)\rangle \\
&- \langle Y(k+1) - Y^*, \beta(\nabla F(Y(k)) - \nabla F(Y^*))\rangle \\
&- \langle \Lambda(k+2) - \Lambda(k+1), Z(k+1)\rangle \\
&- \langle Y(k+1) - Y^*, \beta(\nabla F(Y^*) + (\mathcal{L}\otimes I)\Lambda^*)\rangle
\end{aligned} \tag{23}$$

Since $f(y)$ is smoothly convex function, then we can derive based on the optimal condition [2]:

$$\langle Y(k+1) - Y^*, \beta(\nabla F(Y^*) + (\mathcal{L}\otimes I)\Lambda^*)\rangle \geq 0 \tag{24}$$

By the definition of normal cone, we have:

$$\langle Y(k+1) - Y^*, Z(k+1)\rangle \geq 0 \tag{25}$$

Substituting (23), (24) and (25) back to (20), we obtain:

$$\begin{aligned}
&V(Y(k+1), \Lambda(k+2)) - V(Y(k), \Lambda(k+1)) \\
\leq&- \langle Y(k+1) - Y(k), \mathcal{W}(Y(k+1) - Y(k))\rangle \\
&- \|\Lambda(k+2) - \Lambda(k+1)\|^2 \\
&- 2\langle Y(k+1) - Y^*, (\beta\mathcal{L}\otimes I - 2\beta^2\mathcal{L}^2\otimes I)(Y(k+1) - Y^*)\rangle \\
&- 2\langle Y(k+1) - Y^*, \beta(\nabla F(Y(k)) - \nabla F(Y^*))\rangle
\end{aligned} \tag{26}$$

From Theorem 1, we have $0 < \beta \leq \frac{1}{2\lambda_{max}(\mathcal{L})}$. Moreover, $\mathcal{L}$ is symmetric with a zero eigenvalue, and therefore, we could find an orthogonal matrix $\mathcal{P}$ that $\mathcal{P}^T\mathcal{L}\mathcal{P} = diag\{0, \lambda_1, \cdots, \lambda_N\}$ and $\mathcal{P}^T\mathcal{L}^2\mathcal{P} = diag\{0, \lambda_1^2, \cdots, \lambda_N^2\}$. Then, the matrix $\beta\mathcal{L} - 2\beta^2\mathcal{L}^2$ is positive semi-definite.

Furthermore, the cost function is Lipschitz continuous:

$$\langle Y - Y^*, \nabla F(Y) - \nabla F(Y^*)\rangle \geq \frac{1}{L}\|\nabla F(Y) - \nabla F(Y^*)\|^2 \tag{27}$$

Applying Jensen's inequality to the last term of (26):

$$\begin{aligned}
&- \langle Y(k+1) - Y^*, \nabla F(Y(k)) - \nabla F(Y^*)\rangle \\
=&- \langle Y(k) - Y^*, \nabla F(Y(k)) - \nabla F(Y^*)) \\
&+ \langle -Y(k+1) + Y(k), \nabla F(Y(k)) - \nabla F(Y^*)) \\
\leq&- \frac{1}{L}\|\nabla F(Y(k)) - \nabla F(Y^*)\|^2 + \frac{L}{4}\|Y(k) - Y(k+1)\|^2 \\
&+ \frac{1}{L}\|\nabla F(Y(k)) - \nabla F(Y^*)\|^2 \\
\leq&\frac{L}{4}\|Y(k) - Y(k+1)\|^2
\end{aligned} \tag{28}$$

Then the Equation (26) can be reformed as:

$$\begin{aligned}
&V(Y(k+1), \Lambda(k+2)) - V(Y(k), \Lambda(k+1)) \\
\leq&- \langle Y(k+1) - Y(k), (\mathcal{W} - \frac{\beta L}{2}I)(Y(k+1) - Y(k))\rangle \\
&- \|\Lambda(k+2) - \Lambda(k+1)\|^2
\end{aligned} \tag{29}$$

With Theorem 1, we have $0 < \beta \leq \frac{3}{2L}$, which means $1 - \beta\lambda + \beta^2\lambda^2 - \frac{\beta L}{2} = (\frac{1}{2} - \beta\lambda)^2 + \frac{3}{4} - \frac{\beta L}{2} > 0$. Therefore, $(\mathcal{W} - \frac{\beta L}{2}I)$ is positive definite. In consequence, $V(Y(k+1), \Lambda(k+2)) \leq V(Y(k), \Lambda(k+1))$. Thus, we can conclude that $V(Y, \Lambda)$ converges under the condition of Theorem 1. □

4.3.3 . Before presenting the main result for the distributed optimal cooperative control for general linear systems, we need to apply a state transformation to (8) by letting $x_{i,s}(k) = \Psi_i \xi_i(k)$, $u_{i,s}(k) = G_i \xi_i(k)$. Let $\bar{x}_i(k) = x_i(k) - x_{i,s}(k)$ and $\bar{u}_i(k) = u_i(k) - u_{i,s}(k)$. Applying the control input (8), we have the closed-loop dynamics

$$\begin{aligned} \bar{x}_i(k+1) &= (A_i - B_i K_i)\bar{x}_i(k) - \Psi_i v_i(k) \\ e_i(k) &= C_i \bar{x}_i(k). \end{aligned} \tag{30}$$

The following lemma can be obtained, which can be regarded as input-to-output stability.

Lemma 2. *Let Assumptions 1 and 2 hold. If K_i is chosen such that $A_i - B_i K_i$ is Schur stable and the G_i and Ψ_i are designed by solving the regulation Equations in* (9)*, then there exists a positive constant $\alpha > 0$ such that*

$$\limsup_{k\to\infty} \|e_i(k)\| \le \alpha \limsup_{k\to\infty} \|v_i(k)\|. \tag{31}$$

Proof. First, we examine the solvability of (9). Encapsulating (9) into a matrix form leads to

$$\begin{bmatrix} A_i - I & B_i \\ C_i & 0 \end{bmatrix} \begin{bmatrix} \Psi_i \\ G_i \end{bmatrix} = \begin{bmatrix} 0 \\ I \end{bmatrix}. \tag{32}$$

Denoting $O_i = \begin{bmatrix} A_i - I & B_i \\ C_i & 0 \end{bmatrix}$ and $T_i = \begin{bmatrix} \Psi_i \\ G_i \end{bmatrix}$, by leveraging the property of Kronecker product, $\text{vec}\,(O_i T_i I) = (I \otimes O_i)\,\text{vec}(T_i)$, we can obtain a standard linear algebraic equation

$$(I \otimes O_i)\,\text{vec}(T_i) = \text{vec}\left(\begin{bmatrix} 0 \\ I \end{bmatrix}\right) \tag{33}$$

of which the solvability is guaranteed under (10) in Assumption 2.

For notational convenience, we denote $A_{i,c} = A_i - B_i K_i$ and $B_{i,c} = -\Psi_i$. Then, we have

$$\bar{x}_i(k+1) = A_{i,c}\bar{x}_i(k) + B_{i,c} v_i(k). \tag{34}$$

Recursively iterating (34) results in

$$\bar{x}_i(k) = A_{i,c}^k \bar{x}_i(0) + \sum_{j=0}^{k-1} A_{i,c}^{k-j-1} B_{i,c} v_i(j). \tag{35}$$

Hence, we have

$$e_i(k) = C_i \bar{x}(k) = C_i A_{i,c}^k \bar{x}_i(0) - \sum_{j=0}^{k-1} A_{i,c}^{k-j-1} v_i(j) \tag{36}$$

where $C_i \Psi_i - I = 0$ has been used. Because $A_{i,c}$ is Schur, we have $\lim_{k\to\infty} C_i A_{i,c}^k \bar{x}_i(0) = 0$. In Theorem 1, the convergence of reference generator has been established, which implies v_i converges to zero as $k \to \infty$. Denote $\varpi_i := \limsup_{k\to\infty} \|v_i(k)\|$. Then, for any small constant $\epsilon > 0$, there exists a positive time index $\zeta > 0$ such that

$$\|v_i(k)\| < \varpi_i + \epsilon, \ \forall k > \zeta. \tag{37}$$

Based on the time index ζ, the second term in (36) can be separated into two parts, written as

$$\sum_{j=0}^{k-1} A_{i,c}^{k-j-1} v_i(j) = \sum_{j=0}^{\zeta} A_{i,c}^{k-j-1} v_i(j) + \sum_{j=\zeta+1}^{k-1} A_{i,c}^{k-j-1} v_i(j). \tag{38}$$

Taking the Euclidean norm of (38) and invoking (37):

$$\begin{aligned} \left\|\sum_{j=0}^{k-1} A_{i,c}^{k-j-1} v_i(j)\right\| &= \left\|\sum_{j=0}^{\zeta} A_{i,c}^{k-j-1} v_i(j) + \sum_{j=\zeta+1}^{k-1} A_{i,c}^{k-j-1} v_i(j)\right\| \\ &\le \left\|A_{i,c}^{k-\zeta-1}\right\| \left\|\sum_{j=0}^{\zeta} A_{i,c}^{\zeta-j} v_i(j)\right\| + (\varpi_i + \epsilon)\left\|\sum_{j=\zeta+1}^{k-1} A_{i,c}^{k-j-1}\right\|. \end{aligned} \tag{39}$$

Therefore, we have

$$\limsup_{k\to\infty} \|e_i(k)\| \le \frac{1}{1 - \|A_{i,c}\|}(\varpi_i + \varepsilon) \tag{40}$$

where the following two results have been applied

$$\sum_{j=\zeta+1}^{t-1} \|A_{i,c}\|^{t-1-j} = \frac{1 - \|A_{i,c}\|^{t-\zeta}}{1 - \|A_{i,c}\|} < \frac{1}{1 - \|A_{i,c}\|} \tag{41}$$

$$\lim_{k\to\infty} \left\|A_{i,c}^{k-\zeta-1}\right\| = 0. \tag{42}$$

As ϵ can be set arbitrarily small, it follows from (40) that

$$\limsup_{k\to\infty} \|e_i(k)\| \le \alpha \limsup_{k\to\infty} \|v_i(k)\|. \tag{43}$$

where $\alpha = \frac{1}{1 - \|A_{i,c}\|}$. □

Theorem 2. *Let Assumptions 1 and 2 hold. If K_i are chosen such that $A_i - B_i K_i$ is Schur stable and the G_i and Ψ_i are designed by solving the regulation equations in* (9)*, then the proposed algorithm in* (8) *solves the optimal coordination problem in* (1).

Proof. Let $\tilde{x}_i(k) = x_i(k) - \Psi_i y^*$, we have

$$\begin{aligned} \tilde{x}_i(k+1) =& A_i x_i(k) + B_i[-K_i x_i(k) + (G_i + K_i \Psi_i)\xi_i(k)] - \Psi_i y^* \\ =& (A_i - B_i K_i)\tilde{x}_i(k) + B_i(G_i + K_i \Psi_i)(\xi_i(k) - y^*). \end{aligned} \tag{44}$$

It follows from Theorem 1 and Lemma 2 that $\xi_i(k)$ converges to y^*. Thus, we can conclude the convergence of the proposed algorithm (8) by treating $B_i(G_i + K_i\Psi_i)(\xi_i(k) - y^*)$ as $v_i(k)$ in Lemma 2. □

It is worth mentioning that recursive updating algorithms usually have the format of $y_i(k+1) = y_i(k) + v_i(k)$, where v_i is the change of $y_i(k)$ at time step k. In this paper, we started with this type of integrator dynamics, and then we extended our algorithm to general linear systems by using the classic internal model approach in output regulation where the reference generator has the same updating mechanism as a single integrator.

Remark 3. *In view of the distributed cooperative control literature, e.g., [27], the classic consensus control problem can be regarded as a special case of the optimal cooperative optimisation problems studied in this paper by setting the cost functions as $f_i(y) = (y - y_i(0))^2$. The research problem considered in this paper is initialisation-independent in the sense that the global optimal solution is obtained by solving a distributed optimal coordination problem.*

Remark 4. *In this remark, we discuss the scalability of the solutions. The proposed algorithm is gradient-based, which is easy to compute. The agents only need to communicate with their neighbours. The communication complexity for every agent is $O(n|E|)$, where E is the set of communication edges and n is the number of iterations, because in every iteration the agent only needs to send one message to their neighbours and every message is of constant size $O(1)$. The computational complexity for every agent is also $O(n|E|)$, since every iteration the agent can process the received messages and update the local state in a linear way.*

5 EXPERIMENT RESULTS

This section presents the experimental results to evaluate the effectiveness of the proposed distributed optimisation algorithm. In the beginning, numerical case studies are presented to test the proposed algorithm on multi-robots with linear systems. Then, the proposed algorithm is validated and applied on real-world Turtlebot robots, where all source code, distributed algorithm details and environment setting files are publicly available at our project website upon acceptance.

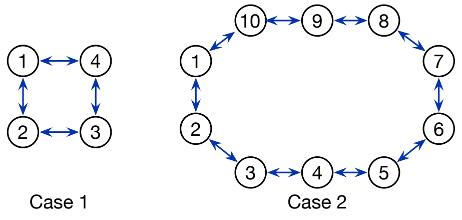

Figure 1: Ring communication topologies for a group of four and ten agents, respectively.

5.1 Numerical Simulation

We test the algorithm on two multi-robot systems consisting of four and ten robots, respectively. Each robot only communicates with its neighbouring robot, and the connected graphs are shown in Fig. 1. For demonstration, we implement the algorithm for a group of four agents (case A), and then extend it to a network of ten agents for the scalability test of the proposed distributed algorithm (case B).

Case A. The agents dynamics are specified as $A = [0, 1; 2, 1]$, $B = [1, 0; 0, 1]$ and $C = [1, 0; 0, 1]$. The assumptions on the graph connectivity, controllablity and regulation conditions are all met. The references of the agents are set as $r_1 = [10; 1], r_2 = [5; 10]; r_3 = [10; 2]; r_4 = [3; 5]$.

The simulation results are shown in Fig. 2. The top left figure is the top view of agents' trajectories. The black points are the initial positions of the four agents, and red points are the final positions of four agents. It can be seen that the agents are driven to the same optimal location which is independent of their initial states but related to the optimisation solution of the references specified. To illustrate more clearly, the convergence processes are shown in the rest three sub-figures, where the top right figure shows the convergence of the intermediate variable λ_i and the bottom figures show the details of the position states x_i and y_i, respectively. From Fig. 2, we can see that the agents reach the consistent goal and the parameters are converged.

5.2 Real Multi-Robot Systems

Case B. We further examine the scalability of the proposed algorithm by implementing it for a set of ten agents, where each agent only needs to communicate and cooperate with its adjacent neighbours. This implies that expanding the size of the network does not incur any additional communication and computation burden, which is one of the advantageous features of distributed control methods.

In addition, we consider a scenario in which the robots' targets change over time. For example, when the robots are controlled to the first target position based on the initial observations, they

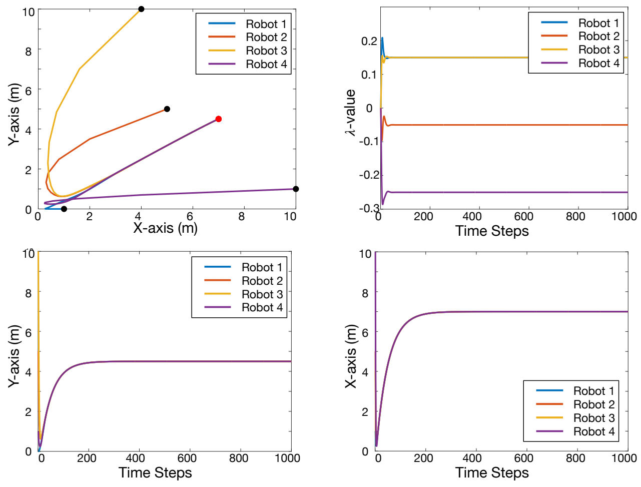

Figure 2: Simulation results for a group of four agents with static local tracking references.

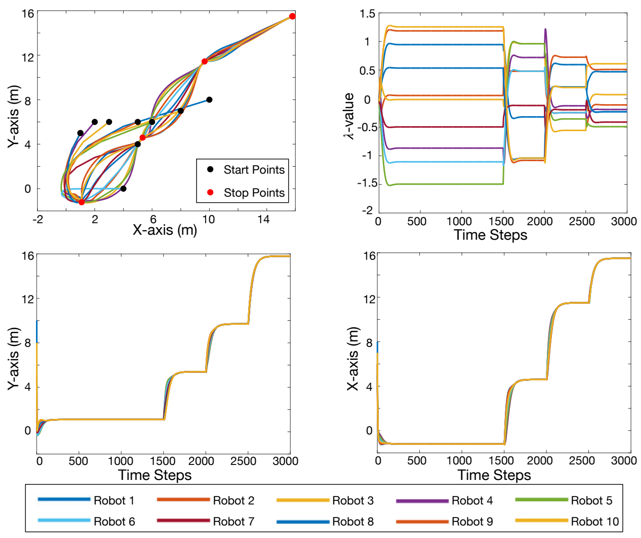

Figure 3: Simulation results for a group of ten agents with rescheduled tracking targets.

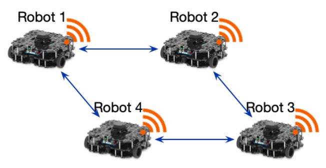

Figure 4: Communication graph for the Turtlebot network.

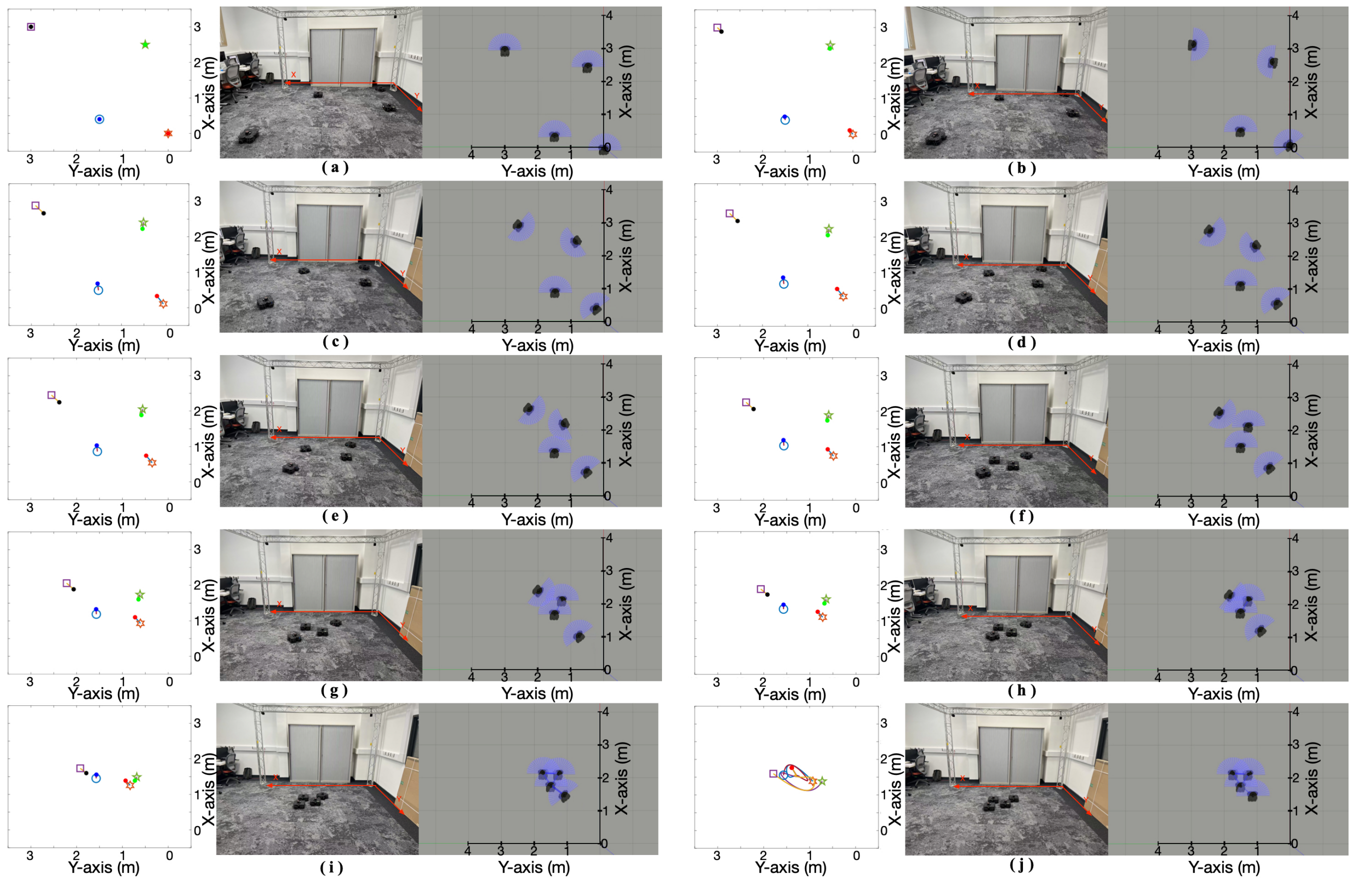

Figure 5: Experiment results on real multi-robot systems.

may re-scan the environment and make a new goal orientation. Therefore, we assume that the robots would re-observe their targets at 1500, 2000, and 2500 steps.

From the simulation results in Fig. 3, it is observed that the proposed algorithm can optimally track the global optimal target no matter where and when the agents re-position their local targets. Therefore, this case study shows both the scalability and initialisation-independent properties of the proposed algorithm.

In this section, we apply the proposed distributed algorithm to the simulation and the physical environments. The robot simulation is based on the ROS and Gazebo platforms. For the physical environment, we assemble four Turtlebot3 Waffle Pi robots, together with a laboratory environment, as shown in the middle of sub-figures in Fig. 5. The Turtlebot3 robots can only get information and communicate with their neighbours, with the communication graph shown in Fig. 4. They are all driven based on the standard ROS platform. We applied the distributed optimisation algorithm to generate an optimal position for every time point. The underlying PID controller will drive the robot to follow the optimal position based on the error between the current and optimal positions. After the top-level optimisation algorithm publishes an optimised reference, the low-level PID controller will trigger two motors to track the reference signal based on Raspberry Pi and OpenCR.

In this experiment, the environment is limited to the laboratory room size, and therefore we design the simulation and real environment within a $3.3 \times 3.5\text{m}^2$ rectangular space. We limited the turning speed of the robots between $[-0.3, 0.3]$ rad/s and the linear speed between $[0, 0.1]$ m/s. The control parameter β is set to 0.05. All the robots know their own initial positions under the same $/world$ frame. The target of the robots are set to $r_1 = [1; 0.1], r_2 = [0.5; 1]; r_3 = [1; 0.2]; r_4 = [0.3; 0.5]$.

The sub-figures (a) to (j) are the exemplar moments during the consensus process, and a recorded video will be available upon acceptance. In each sub-figure, the left simulated graph provides the high-level reference that is generated by the proposed distributed optimisation algorithm, while the right two figures show the tracking performances of the simulated and real robots. It is noted that the real robots approach the optimised reference generated by cooperative optimisation instead of their local target, and the stopping distance is set as 0.3m to avoid the collision of the real robots.

In the real-world experiment, there are errors in the robot system dynamic information (position) of the vehicle due to the interference of external factors, such as different ground friction. Nevertheless, the proposed algorithm can deal with them and update its local information by communicating with adjacent agents to optimise the global cost. From the figures and video, we observe that the proposed distributed cooperative control algorithm always optimises the global optimal position for each robot. The robots reach the consensus of the final target by the proposed approach. The problem under our consideration covers a wider range of cooperative control problems, including initialisation-dependent consensus problems, and collaborative target tracking and search problems. The deployment and realisation of the proposed algorithm on real robotic systems demonstrate its significant potential for real-world applications.

6 CONCLUSIONS

This paper proposes an output regulation-based distributed optimisation algorithm in the context of multi-agent cooperative control. It optimises the local objectives of the agents by letting them communicating with their neighbours, and in the meantime ensures that the agents can reach the global optimal. The algorithm can track the global optimal target, instead of converging to a centre based on initial positions, and moreover, it can handle multi-agent systems with linear and heterogeneous dynamics. Both theoretical guarantee and experimental validations are studies, which will promote the future deployment of distributed cooperative optimal control to solve real world applications with guaranteed convergence and optimality.

ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 956123, and is also supported by the UK EPSRC under project [EP/T026995/1].

REFERENCES

[1] Ilario Antonio Azzollini, Nicola Mimmo, Lorenzo Gentilini, and Lorenzo Marconi. 2021. UAV-Based Search and Rescue in Avalanches using ARVA: An Extremum Seeking Approach. *arXiv preprint arXiv:2106.14514* (2021).

[2] Dimitri Bertsekas. 2009. *Convex optimization theory*. Vol. 1. Athena Scientific.

[3] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[4] Chi-Tsong Chen. 1984. *Linear system theory and design*. Saunders college publishing.

[5] Ashish Cherukuri and Jorge Cortes. 2016. Initialization-free distributed coordination for economic dispatch under varying loads and generator commitment. *Automatica* 74 (2016), 183–193.

[6] Zhengtao Ding. 2003. Global stabilization and disturbance suppression of a class of nonlinear systems with uncertain internal model. *Automatica* 39, 3 (2003), 471–479.

[7] Zhengtao Ding. 2013. Consensus output regulation of a class of heterogeneous nonlinear systems. *IEEE Trans. Automat. Control* 58, 10 (2013), 2648–2653.

[8] Jie Huang. 2004. *Nonlinear output regulation: theory and applications*. SIAM.

[9] Yasuhiro Kuriki and Toru Namerikawa. 2014. Consensus-based cooperative formation control with collision avoidance for a multi-UAV system. In *2014 American Control Conference*. IEee, 2077–2082.

[10] Jinlong Lei, Han-Fu Chen, and Hai-Tao Fang. 2016. Primal–dual algorithm for distributed constrained optimization. *Systems & Control Letters* 96 (2016), 110–117.

[11] Li Li, Yang Yu, Xiuxian Li, and Lihua Xie. 2022. Exponential convergence of distributed optimization for heterogeneous linear multi-agent systems over unbalanced digraphs. *Automatica* 141 (2022), 110259.

[12] Zhongguo Li. 2021. *Distributed Cooperative and Competitive Optimisation Over Networks*. The University of Manchester (United Kingdom).

[13] Zhongguo Li, Wen-Hua Chen, and Jun Yang. 2021. Concurrent Learning Based Dual Control for Exploration and Exploitation in Autonomous Search. *arXiv preprint arXiv:2108.08062* (2021).

[14] Zhongguo Li, Zhen Dong, Zhongchao Liang, and Zhengtao Ding. 2021. Surrogate-based distributed optimisation for expensive black-box functions. *Automatica* 125 (2021), 109407. https://doi.org/10.1016/j.automatica.2020.109407

[15] Zhongkui Li, Zhisheng Duan, and Guanrong Chen. 2011. Consensus of discrete-time linear multi-agent systems with observer-type protocols. *arXiv preprint arXiv:1102.5599* (2011).

[16] Luka Martinović, Žarko Zečević, and Božo Krstajić. 2022. Cooperative tracking control of single-integrator multi-agent systems with multiple leaders. *European Journal of Control* 63 (2022), 232–239.

[17] Angelia Nedić and Alex Olshevsky. 2014. Distributed optimization over time-varying directed graphs. *IEEE Trans. Automat. Control* 60, 3 (2014), 601–615.

[18] Angelia Nedic and Asuman Ozdaglar. 2009. Distributed subgradient methods for multi-agent optimization. *IEEE Trans. Automat. Control* 54, 1 (2009), 48–61.

[19] Angelia Nedic, Asuman Ozdaglar, and Pablo A Parrilo. 2010. Constrained consensus and optimization in multi-agent networks. *IEEE Trans. Automat. Control* 55, 4 (2010), 922–938.

[20] Boda Ning, Qing-Long Han, and Zongyu Zuo. 2017. Distributed optimization for multiagent systems: An edge-based fixed-time consensus approach. *IEEE Transactions on Cybernetics* 49, 1 (2017), 122–132.

[21] Boda Ning, Qing-Long Han, and Zongyu Zuo. 2020. Bipartite consensus tracking for second-order multiagent systems: A time-varying function-based preset-time approach. *IEEE Trans. Automat. Control* 66, 6 (2020), 2739–2745.

[22] Reza Olfati-Saber and Richard M Murray. 2004. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Automat. Control* 49, 9 (2004), 1520–1533.

[23] Edwin Olson. 2011. AprilTag: A robust and flexible visual fiducial system. In *2011 IEEE international conference on robotics and automation*. IEEE, 3400–3407.

[24] Minkyu Park and Hyondong Oh. 2020. Cooperative information-driven source search and estimation for multiple agents. *Information Fusion* 54 (2020), 72–84.

[25] Jiahu Qin, Qichao Ma, Yang Shi, and Long Wang. 2016. Recent advances in consensus of multi-agent systems: A brief survey. *IEEE Transactions on Industrial Electronics* 64, 6 (2016), 4972–4983.

[26] Zhirong Qiu, Shuai Liu, and Lihua Xie. 2016. Distributed constrained optimal consensus of multi-agent systems. *Automatica* 68 (2016), 209–215.

[27] Zhihua Qu. 2009. *Cooperative control of dynamical systems: applications to autonomous vehicles*. Springer Science & Business Media.

[28] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[29] Wei Ren and Randal W Beard. 2005. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Trans. Automat. Control* 50, 5 (2005), 655–661.

[30] Wei Ren, Randal W Beard, and Ella M Atkins. 2007. Information consensus in multivehicle cooperative control. *IEEE Control systems magazine* 27, 2 (2007), 71–82.

[31] Wei Ren and Dimos V Dimarogonas. 2022. Event-Triggered Tracking Control of Networked Multi-Agent Systems. *IEEE Trans. Automat. Control* (2022).

[32] Branko Ristic, Christopher Gilliam, William Moran, and Jennifer L. Palmer. 2020. Decentralised multi-platform search for a hazardous source in a turbulent flow. *Information Fusion* 58 (2020), 13–23.

[33] Guodong Shi and Yiguang Hong. 2009. Global target aggregation and state agreement of nonlinear multi-agent systems with switching topologies. *Automatica* 45, 5 (2009), 1165–1175.

[34] Enrica Soria, Fabrizio Schiano, and Dario Floreano. 2021. Distributed Predictive Drone Swarms in Cluttered Environments. *IEEE Robotics and Automation Letters* 7, 1 (2021), 73–80.

[35] Enrica Soria, Fabrizio Schiano, and Dario Floreano. 2021. Predictive control of aerial swarms in cluttered environments. *Nature Machine Intelligence* 3, 6 (2021), 545–554.

[36] Ngoc-Tu Tran, Yan-Wu Wang, Xiao-Kang Liu, Jiang-Wen Xiao, and Yan Lei. 2019. Distributed optimization problem for second-order multi-agent systems with event-triggered and time-triggered communication. *Journal of the Franklin Institute* 356, 17 (2019), 10196–10215.

[37] John Wang and Edwin Olson. 2016. AprilTag 2: Efficient and robust fiducial detection. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4193–4198.

[38] Xiaoli Wang, Yiguang Hong, Jie Huang, and Zhong-Ping Jiang. 2010. A distributed control approach to a robust output regulation problem for multi-agent linear systems. *IEEE Transactions on Automatic control* 55, 12 (2010), 2891–2895.

[39] Xinghu Wang, Yiguang Hong, and Haibo Ji. 2015. Distributed optimization for a class of nonlinear multiagent systems with disturbance rejection. *IEEE Transactions on Cybernetics* 46, 7 (2015), 1655–1666.

[40] Yujuan Wang, Yongduan Song, David J Hill, and Miroslav Krstic. 2018. Prescribed-time consensus and containment control of networked multiagent systems. *IEEE Transactions on Cybernetics* 49, 4 (2018), 1138–1147.

[41] Tao Yang, Xinlei Yi, Junfeng Wu, Ye Yuan, Di Wu, Ziyang Meng, Yiguang Hong, Hong Wang, Zongli Lin, and Karl H Johansson. 2019. A survey of distributed optimization. *Annual Reviews in Control* 47 (2019), 278–305.

[42] Peng Yi, Yiguang Hong, and Feng Liu. 2016. Initialization-free distributed algorithms for optimal resource allocation with feasibility constraints and application

to economic dispatch of power systems. *Automatica* 74 (2016), 259–269.

[43] Yu Zhao, Yongfang Liu, Guanghui Wen, and Guanrong Chen. 2017. Distributed optimization for linear multiagent systems: Edge-and node-based adaptive designs. *IEEE Trans. Automat. Control* 62, 7 (2017), 3602–3609.

[44] Yao Zou, Zongyu Zuo, and Kewei Xia. 2021. Sampled-data distributed protocol for coordinated aggregation of multi-agent systems subject to communication delays. *Nonlinear Analysis: Hybrid Systems* 43 (2021), 101108.

[45] Shan Zuo, Yongduan Song, Frank L Lewis, and Ali Davoudi. 2018. Adaptive output formation-tracking of heterogeneous multi-agent systems using time-varying $\mathcal{L}_2$-gain design. *IEEE Control Systems Letters* 2, 2 (2018), 236–241.

[46] Zongyu Zuo, Qing-Long Han, and Boda Ning. 2019. *Fixed-time cooperative control of multi-agent systems.* Springer.