Distributed Cooperative Kinematic Control of Multiple Robotic Manipulators with Improved Communication Efficiency

Jiazheng Zhang, Long Jin, *Member, IEEE*, Chenguang Yang, *Senior Member, IEEE*

***Abstract*—An efficiency-oriented solution is theoretically a preferred choice to support the efficient operation of a system. Although some studies on the multi-manipulator system share the load of the control center by transforming the network topology, the whole system often suffers an increased communication burden. In this article, a multi-manipulator cooperative control scheme with improved communication efficiency is proposed to allocate limited communication resources reasonably. The entire control process is formulated from the perspective of game theory and finally evolved into a problem of finding a Nash equilibrium with time-varying parameters. Then, a neural network solver is designed to update the strategies of manipulators. Theoretical analysis supports the convergence and robustness of the solver. In addition, Zeno behavior does not occur under the domination of the control strategy. Finally, simulative results reveal that the proposed control strategy has advantages over traditional periodic control in communication.**

***Index Terms*—Distributed control, neural networks, communication overhead, redundancy resolution, game theory.**

I. Introduction

WITH the development of engineering fields and the promotion of intelligent production, current technologies impose more demands in the improving of traditional centralized control systems or frameworks. Constructing an intelligent system that integrates computation, communication and control technologies has become a rigid demand for promoting industrial development. At present, more and more agent systems are designed with cyber-physical systems as the vane, which aims to coordinate computing, communication and other resources to provide high-quality control for the system [1].

As a unique branch of the robot product family, manipulators have attracted special attention in various fields due to its strong practicality [2]. Over the past half century, various kinds of manipulators have been developed for applications such as minimally invasive surgery [3], space target capture [4], precision assembly [5]. Although a single manipulator satisfies the needs of some industries well in many respects, it lacks certain economic efficiency in some complex or large-scale operations. For this reason, a variety of multi-manipulator systems have been emerging to cope with this deficiency, with an obvious advantage that the tasks are completed efficiently and flexibly through the cooperation of manipulators [6]. In recent developments, the research on multi-manipulator systems mostly focuses on their control strategy formulations, including centralized strategies [7] and distributed strategies [8], [9], intelligent control strategies [10], [11], and graph-based strategies [12], [13], etc. In [7], an execution instruction is generated and sent to each manipulator uniformly by a command center, but a potential defect is that the command center may not be able to dispatch multiple manipulators online in a timely manner, and the reliability of the system is poor. The distributed control strategy is an improvement made to alleviate the operating pressure of the command center, e.g., the control strategy in [8], [9], which amortizes the operating load of the command center by allowing each manipulator to communicate locally. Besides, to handle some uncertain factors that may appear in the control process, a kind of intelligent algorithms such as adaptive control [10] and fuzzy control [11] are presented, which are currently deemed as important means to improve the dynamic performance of manipulators. Apart from these, the topological connection of manipulators determines the performance of the system to perform tasks to a certain extent, so graph-based strategies are also investigated. Typically, differences in performance of various control architectures are revealed in [12] from the perspective of graph theory.

While multi-manipulator systems maintain the advantages of scalability, stability, and efficiency in performing complex tasks, they also bring high control costs [14]. Calculation and communication are two processes that occupy the most resources in system operation [15], [16]. Notice that in traditional multi-manipulator systems, the system controller mostly adopts a time-triggered update method. That is, each solver obtains the required information at the predefined intervals

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFE0118900, in part by the research project of Huawei Mindspore Academic Award Fund of Chinese Association of Artificial Intelligence under Grant CAAIXSJLJJ-2020-009A, in part by the Team Project of Natural Science Foundation of Qinghai Province, China, under Grant 2020-ZJ-903, in part by the Key Laboratory of IoT of Qinghai under Grant 2020-ZJ-Y16, in part by the Natural Science Foundation of Gansu Province, China, under Grant 20JR10RA639, in part by the Natural Science Foundation of Chongqing (China) under Grant cstc2020jcyj-zdxmX0028, in part by the Research and Development Foundation of Nanchong (China) under Grant 20YFZJ0018, in part by CAS "Light of West China" Program, in part by the Project Supported by Chongqing Key Laboratory of Mobile Communications Technology under Grant cqupt-mct-202004, and in part by the Fundamental Research Funds for the Central Universities under Grant lzujbky-2019-89 and Grant lzuxxxy-2019-tm20. *(Corresponding author: Long Jin.)*.

J. Zhang and L. Jin are with the School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China, and also with the Academy of Plateau Science and Sustainability, Xining 810016, China (e-mails: zhangjzh15@foxmail.com; jinlongsysu@foxmail.com).

Chenguang Yang is with Bristol Robotics Laboratory, University of the West of England, Bristol, BS16 1QY, UK (Email: cyang@ieee.org).

through sensors, and then carries out computations and updates [17]. However, a significant defect is that the controller is in continuous operation regardless of whether the system needs to update the strategy, thus ignoring the energy consumption [1], [18], [19]. To this end, it is necessary to design a new system controller, which requires calculation and communication to be started only when the manipulator needs to update the strategy, not at a predetermined moment [20]. For example, a dynamic event-triggered control strategy is designed in [21], which can guide the single-link manipulator to obtain a compromise solution between control accuracy and communication overhead. In [22], an event-triggered condition is set to decide whether information exchange is executed among agents, which significantly reduces the redundant transmission in the wireless network. Moreover, the design concept of triggering communication is applied to formation control of multiple agents in [23], which effectively reduces the communication between leaders as well as followers. Nevertheless, existing strategies with the ability to improve communication efficiency mostly focus on exploring systems that can treat agents as particles, e.g., [24]–[26], and rarely study scenarios involving the posture of manipulators. Therefore, constructing a multi-manipulator control strategy with improved communication efficiency and position consideration has potential application value in industrial production.

Neural network solver is an efficient and practical tool widely used in manipulator control at present, with powerful real-time processing capabilities [27]–[30]. For example, a neural network is applied to the real-time tracking of a wheeled manipulator in [28] and achieves the expected control effect. Beyond that, solvers designed based on neural networks have attractive prospects in the development of multi-manipulator systems. For instance, a problem with the goal of optimizing the manipulability of the manipulator in the system is solved online by a neural network solver in [31]. In [32], a neural network solver is designed to plan the optimal kinematic scheme that can make the manipulator operation stably. In view of the unique performance advantages of neural network in computing, it can be rationally applied to the control scheme design of the manipulator system. In this article, a multi-manipulator cooperative control scheme with improved communication efficiency is constructed from a game perspective. The strategy of each manipulator is designed to be updated with properly defined trigger conditions. To the best of our knowledge, there is no systematic solution on recurrent neural network design for multi-manipulator cooperation with improved system communication efficiency.

The rest of this article is organized as follows. Section II presents some preliminaries and uses the game theory frame to formulate the cooperation problem of multiple manipulators. Then, Section III introduces the event-triggered mechanism into the multi-manipulator system, and proposes a neural network solver for real-time updating of control strategies. In Section IV, theoretical analysis is given to support the performance of the proposed solver. Besides, simulation results are illustrated in Section V. Finally, Section VI concludes the article.

II. Preliminaries and Problem Formulation

In this section, the kinematic equations of the manipulator and some basic graph theories are given as preliminaries. Moreover, a distributed cooperative control of multiple manipulators is formulated in the framework of game theory.

A. Robot Manipulator Kinematics

In the kinematic control of the manipulator, the spatial position of the end-effector is uniquely determined by the state of the joints. To describe this correspondence between joint space $\vartheta(t) \in \mathbb{R}^m$ and Cartesian space $\mathbf{x}(t) \in \mathbb{R}^n$, it is conventionally defined as

$$\mathbf{x}(t) = \mathsf{M}\big(\vartheta(t)\big), \tag{1}$$

where $\mathsf{M}(\cdot) : \mathbb{R}^m \to \mathbb{R}^n$, $\mathbf{x}(t) = [x^1(t); x^2(t); \ldots; x^n(t)]$, and $\vartheta(t) = [\theta^1(t); \theta^2(t); \ldots; \theta^m(t)]$. Furthermore, for any manipulator with known structural information, (1) can be evolved into the following affine system:

$$\dot{\mathbf{x}}(t) = \mathcal{J}\big(\vartheta(t)\big)\dot{\vartheta}(t), \tag{2}$$

with $\dot{\mathbf{x}}(t) = \partial \mathbf{x}(t)/\partial t \in \mathbb{R}^n$, $\mathcal{J}\big(\vartheta(t)\big) = \partial \mathsf{M}\big(\vartheta(t)\big)/\partial \vartheta \in \mathbb{R}^{n\times m}$, and $\dot{\vartheta}(t) = \partial \vartheta(t)/\partial t \in \mathbb{R}^m$, which emphasizes the instantaneous kinematic behavior of the manipulator. In fact, how to deduce the rotation status of each joint through the real-time position of the end-effector has always been a concern in manipulator control. In this article, the desired trajectory of the end-effector is set to $\mathbf{x}_{\mathrm{d}}(t)$, which is emitted from the command center.

B. Graph Theory

In a system with S manipulators, the communication links among the manipulators can be stored in graph $\mathcal{G} \triangleq (\mathsf{V}, \mathsf{E})$, where $\mathsf{V} = \{1, 2, \ldots, S\}$ refers to the vertex set with S manipulators, and $\mathsf{E} \subseteq \mathsf{V}\times\mathsf{V}$ refers to the edge set that records the connected relation of the manipulators. For manipulator i, the remaining manipulators covered in its communication range constitute the set $\mathsf{N}_i = \{j \in \mathsf{V} : (i, j) \in \mathsf{E}\}$. To further represent the connection relationship between any manipulator i and j, its adjacency matrix is set as

$$\mathcal{A} = [\alpha_{ij}] \in \mathbb{R}^{S\times S} : \alpha_{ii} = 0,\ \alpha_{ij} = \begin{cases} 1, & \text{if } (i, j) \in \mathsf{E} \\ 0. & \text{otherwise} \end{cases}$$

Further, the Laplace matrix of $\mathcal{G}$ can be defined as

$$\mathsf{L} = \mathrm{diag}(\mathcal{A}\mathbf{e}_S) - \mathcal{A} \in \mathbb{R}^{S\times S},$$

where $\mathbf{e}_S = [1; 1; \ldots; 1] \in \mathbb{R}^S$. In addition, a state vector $\phi = [\phi_i] \in \mathbb{R}^S$ is designed to record the connection between the manipulator and the command center, in which $\phi_i = 1$ if the communication is established, and $\phi_i = 0$ otherwise.

Assumption 1: $\mathcal{G}$ is assumed to be strongly connected in this article, and only a limited number of manipulators in the system can directly receive commands from the command center.

C. Game-Theoretic Formulation of Problem

Consider a scenario in which a manipulator system with limited communication resources is executing payload transport task. At this point, the centralized communication mode is no longer useful. An alternative solution is to give each manipulator the power of small-scale communication, enabling them to communicate and cooperate via a network to reduce the overhead of the control center. However, note that the instructions to drive the manipulators in this alternative are no longer pre-generated by the command center, so the general optimization framework is not suitable for solving this problem. Given that each manipulator determines its own behavior by accessing the information from nearby manipulators, its utility function depends not only on its own decisions, but also on those of other manipulators. With this in mind, a game theory framework can be adopted to deal with the problem. Here, the manipulators are regarded as game players. Stimulated by rewards or punishments in the game, the manipulators will instinctively make decisions that benefit themselves.

For those strategies that can guide the manipulator to complete the assigned tasks, their performance on the manipulator has significant disparities. To quantify the performance gain of the action acting on the manipulator at each moment, a payoff function in a quadratic form is defined as

$$\mathsf{P}(t) = -\big(\dot{\vartheta}^\top(t)\mathcal{R}\dot{\vartheta}(t)/2 + \mathbf{p}^\top\dot{\vartheta}(t)\big),$$

where $(\cdot)^\top$ denotes the transpose of a matrix or a vector; $\mathcal{R} \in \mathbb{R}^{m\times m}$ is a semi-positive definite matrix; $\mathbf{p} \in \mathbb{R}^m$ is a vector. Actually, the main problem of multi-manipulator game is the disposal of the resource paradox and the allocation of finite resources. It is conceivable that when each manipulator tries to maximize its own benefits from finite resources, a conflict between local interests and overall interests arises. Therefore, it is necessary to formulate corresponding game rules to control the behavior of each manipulator. For this, the strategy of the ith manipulator in the cooperative task can be generated according to the following rules:

$$\max \quad \mathsf{P}_i(t) = -\big(\dot{\vartheta}_i^\top(t)\mathcal{R}\dot{\vartheta}_i(t)/2 + \mathbf{p}^\top\dot{\vartheta}_i(t)\big) \tag{3a}$$

$$\text{s.t.} \quad \dot{\mathbf{x}}_i(t) = \mathcal{J}_i\big(\vartheta_i(t)\big)\dot{\vartheta}_i(t) \tag{3b}$$

$$\mathsf{X}_i^\circ = \frac{1}{\sum_{j\in\mathsf{N}_i}\alpha_{ij}}\sum_{j\in\mathsf{N}_i}\alpha_{ij}\mathsf{X}_j^\circ \tag{3c}$$

$$\text{with } \mathsf{X}_j^\circ = \begin{cases}\aleph_j(t), & \text{if } \phi_j = 0\\ \mathbf{x}_{\mathrm{d}}(t), & \text{if } \phi_j = 1\end{cases}$$

of which $\aleph_j(t) = \mathbf{x}_j(t) - \flat_j^{\mathsf{c}}$ with $\flat_j^{\mathsf{c}}$ is a constant vector of jth end-effector to a fixed point of the payload. Emphasize that, the existence of (3a) is to maximize the expected payoff of the manipulator, which is one of the key considerations in this game; (3b) is the Jacobian equation constraint of the manipulator i; furthermore, (3c) and (3d) are the information acquisition constraints that every player is required to abide by in the game. Particularly, constraints (3c) and (3d) stipulate that the manipulator not connected to the control center determines its motion behavior by using the weighted average of surrounding manipulators' X°, which are not allowed to be broken without authorization.

Remark 1: To reasonably import the manipulator model into the designed information acquisition constraints, we make a preliminary integration of (3b), (3c) and (3d). First, combine (3c) and (3d) to obtain

$$\sum_{j\in\mathsf{N}_i}\alpha_{ij}(\mathsf{X}_i^\circ - \mathsf{X}_j^\circ) + \sum_{j\in\mathsf{N}_i}\phi_i(\mathsf{X}_i^\circ - \mathbf{x}_{\mathrm{d}}(t)) = 0. \tag{4}$$

Subsequently, an error function of (4) is defined as

$$\hbar_i = \sum_{j\in\mathsf{N}_i}\alpha_{ij}(\mathsf{X}_i^\circ - \mathsf{X}_j^\circ) + \sum_{j\in\mathsf{N}_i}\phi_i(\mathsf{X}_i^\circ - \mathbf{x}_{\mathrm{d}}(t)) - 0. \tag{5}$$

Further, by means of $\gamma\dot{\hbar}_i = -\hbar_i$ with $\gamma > 0$ and combining (3b), (4) can be equivalently converted to

$$\begin{aligned}&\gamma\sum_{j\in\mathsf{N}_i}\big(\alpha_{ij}(\mathcal{J}_i\dot{\vartheta}_i(t) - \mathcal{J}_j\dot{\vartheta}_j(t)) + \phi_i(\mathcal{J}_i\dot{\vartheta}_i(t) - \dot{\mathbf{x}}_{\mathrm{d}}(t))\big)\\ &+\sum_{j\in\mathsf{N}_i}\big(\alpha_{ij}(\mathsf{X}_i^\circ - \mathsf{X}_j^\circ) + \phi_i(\mathsf{X}_i^\circ - \mathbf{x}_{\mathrm{d}}(t))\big) = 0.\end{aligned} \tag{6}$$

Remarkably, (6) can be reconstructed into the following compact form under the guidance of graph theory:

$$\begin{aligned}&\gamma(\mathsf{L}+\Phi)\otimes I_n\tilde{\mathcal{J}}\dot{\tilde{\vartheta}} + (\Phi\otimes I_n)\big(\mathbf{e}_S\otimes\mathbf{x}_{\mathrm{d}}(t)\big)\\ =&\gamma(\Phi\otimes I_n)\big(\mathbf{e}_S\otimes\dot{\mathbf{x}}_{\mathrm{d}}(t)\big) + (\mathsf{L}+\Phi)\otimes I_n\tilde{\mathsf{X}}^\circ,\end{aligned} \tag{7}$$

of which $\Phi = \mathrm{diag}(\phi) \in \mathbb{R}^{S\times S}$; $\tilde{\mathcal{J}} = \mathrm{diag}(\mathcal{J}_1;\ldots;\mathcal{J}_S) \in \mathbb{R}^{nS\times mS}$; $\dot{\tilde{\vartheta}} = [\dot{\vartheta}_1;\ldots;\dot{\vartheta}_S] \in \mathbb{R}^{mS}$; $\dot{\mathbf{x}}_{\mathrm{d}}(t) = \partial\mathbf{x}_{\mathrm{d}}(t)/\partial t \in \mathbb{R}^n$; $\tilde{\mathsf{X}}^\circ = [\mathsf{X}_1^\circ;\ldots;\mathsf{X}_S^\circ] \in \mathbb{R}^{nS}$; $\otimes$ signifies the Kronecker product. Given the above, the subconstraints (3b), (3c), and (3d) is able to be uniformly replaced by (7).

Hereto, the distributed cooperative task of manipulators with limited communication resources is formally established as a game for S players. Taking into account the mutual restriction brought by manipulators when making decisions, the final result of the game will develop towards equilibrium. In game theory terminology, the combination of strategies that create such an equilibrium is known as Nash equilibrium. Note that once equilibrium is established, there will be no manipulator willing to unilaterally change its strategy, since this is already the best choice the player can make in the game.

III. A New Multi-Robot Cooperation Scheme

Although the system adopting distributed communication greatly weakens the demand for the command center, there are still some unnecessary communication overheads that will be generated in the local information exchange. To this end, this section introduces an event-triggered control mechanism, and finds the Nash equilibrium in the game on this basis.

A. Event-Triggered Conditions

To reduce unnecessary communication consumption and calculation overhead, an event-triggered control mechanism can be adopted to determine whether the manipulator needs to send or update data by observing the change trend of the error. From the perspective of the game theory, if changing the strategy will incur some cost, the above process can be regarded as a trade-off between whether the player maintains

the existing payoff or actively adjusts the strategy to reap a more considerable payoff.

Notably, the position error of manipulator i with $\phi_i = 0$ can be defined as an accumulation of drift errors generated by its end-effector and surrounding end-effectors in a cooperative motion. Specifically, the above-mentioned error is expressed as

$$\varepsilon_i(t) = \sum_{j\in\mathsf{N}_i} \alpha_{ij}\big(\aleph_i(t) - \aleph_j(t)\big). \tag{8}$$

Then, a Lyapunov function is constructed as

$$\mathfrak{L}(\varepsilon_i) = \varepsilon_i^\top(t)\varepsilon_i(t)/2. \tag{9}$$

Evidently, $\mathfrak{L}(\varepsilon_i)$ is positive definite because $\mathfrak{L}(\varepsilon_i) = 0$ only if $\varepsilon_i = 0$, and in other cases $\mathfrak{L}(\varepsilon_i) > 0$. Subsequently, $\dot{\mathfrak{L}}(\varepsilon_i) = \partial\mathfrak{L}(\varepsilon_i)/\partial t = \varepsilon_i^\top(t)\dot{\varepsilon}_i(t)$ can be obtained conveniently. Based on the definitions of $\mathfrak{L}(\varepsilon_i)$ and $\dot{\mathfrak{L}}(\varepsilon_i)$, in order to formulate an event-triggered function in a convenient manner, the inputs between the successive calculation of $\dot{\mathfrak{L}}(\varepsilon_i)$ are assumed to be held constant, i.e., sample-and-hold [33]. Accordingly, the specific equation is designed as

$$\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) = \big(\mathfrak{L}_{t_{k+1}}(\varepsilon_i) - \mathfrak{L}_{t_k}(\varepsilon_i)\big)/(t_{k+1}-t_k) + \lambda\mathfrak{L}_{t_k}(\varepsilon_i),$$

where $\lambda > 0$ is a design parameter, and a greater value of λ leads to a more sensitive $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big)$; sequence $\{t_k\}_{k\in\mathbb{N}}$ denotes the instants at which $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big)$ is recomputed and the control strategy is updated. It is worth stressing that the state of $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big)$ determines whether manipulator i needs to update its own strategy and synchronize end-effector data to the surrounding manipulators. In brief, when $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) > 0$, the manipulator needs to adjust its own strategy and establish communication with the surrounding manipulators; When $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) \le 0$, the manipulator only needs to stay in the previous state without performing the above steps.

Remark 2: $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) \le 0$ is an important criterion for judging whether the manipulator is able to maintain the previous strategy in task execution. Notice that since $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) \le 0$ and $\lambda\mathfrak{L}(\varepsilon_i) \ge 0$, $\dot{\mathfrak{L}}(\varepsilon_i) \le 0$ can be readily obtained. At this time, $\dot{\mathfrak{L}}(\varepsilon_i)$ is negative semi-definite. Moreover, we know that $\mathfrak{L}(\varepsilon_i)$ is positive definite and that $\mathfrak{L}(\varepsilon_i) \to \infty$ when $\varepsilon_i \to \infty$. Therefore, (8) is asymptotically stable under the condition of $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) \le 0$. It can be expected that $\aleph_i(t)$ will eventually converge to $\aleph_j(t)$ over time, which means that the end-effector motion behavior of manipulator i and j will be gradually consistent.

B. Reformulation of Manipulator Control Rules

In the current game, each manipulator updates its own strategic behavior by exercising the right to maximize $\mathsf{P}_i(t)$. Subject to the designed constraints, the game problem is further abstracted as (3), thus mathematically describing the conflict of interests among the manipulators.

To facilitate the solution of (3), we try to transform it into a quadratic programming problem. Furthermore, by adopting (7) and combining with the designed event-triggered conditions, the rules for strategy generation of the system are formulated as below:

$$\begin{aligned}
\min\ & \tilde{\dot{\vartheta}}^\top(t)(I_S\otimes\mathcal{R})\tilde{\dot{\vartheta}}(t)/2 + (\mathbf{e}_S\otimes\mathbf{p})^\top\tilde{\dot{\vartheta}}(t)\\
\text{s.t.}\ & \gamma(\mathsf{L}+\Phi)\otimes I_n\tilde{\mathcal{J}}\tilde{\dot{\vartheta}} + (\Phi\otimes I_n)\big(\mathbf{e}_S\otimes\mathbf{x}_{\mathrm{d}}(t)\big)\\
& = \gamma(\Phi\otimes I_n)\big(\mathbf{e}_S\otimes\dot{\mathbf{x}}_{\mathrm{d}}(t)\big) + (\mathsf{L}+\Phi)\otimes I_n\tilde{\mathrm{X}}^\circ,\\
\text{triggering condition:}\ & \begin{cases}\text{previous strategy}, & \text{if } \mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big)\le 0,\\ \text{updated strategy}, & \text{if } \mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) > 0.\end{cases}
\end{aligned}$$

It can be seen that the manipulators use a minimization index and a constraint to develop their own strategies, and each manipulator updates its own strategy only at $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) > 0$, while maintaining the previous strategy in other cases. In particular, when $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) > 0$, the above optimization problem is equivalent to solving the following equations by means of Lagrange-multiplier method [34]:

$$\begin{cases}(I_S\otimes\mathcal{R})\tilde{\dot{\vartheta}}(t) + \mathbf{e}_S\otimes\mathbf{p} + \gamma\big((\mathsf{L}+\Phi)\otimes I_n\tilde{\mathcal{J}}\big)^\top\kappa(t) = \mathbf{0},\\ \gamma(\mathsf{L}+\Phi)\otimes I_n\tilde{\mathcal{J}}\tilde{\dot{\vartheta}} + \omega(t) = \mathbf{0},\end{cases} \tag{10}$$

where $\kappa(t) = [\kappa_1(t);\ldots;\kappa_S(t)] \in \mathbb{R}^{nS}$ is the Lagrange multiplier vector with $\kappa_i \in \mathbb{R}^n$, and $\omega(t) = (\Phi\otimes I_n)\big(\mathbf{e}_S\otimes\mathbf{x}_{\mathrm{d}}(t) - \gamma\mathbf{e}_S\otimes\dot{\mathbf{x}}_{\mathrm{d}}(t)\big) - (\mathsf{L}+\Phi)\otimes I_n\tilde{\mathrm{X}}^\circ$. Further, to facilitate monitoring the status updates of the strategy $\tilde{\dot{\vartheta}}(t)$ of the manipulators, (10) is reconstructed as follows:

$$\Pi(t)\begin{bmatrix}\mathbf{u}_{\dot{\vartheta}}(t)\\ \mathbf{u}_{\kappa}(t)\end{bmatrix} - \Psi(t) = 0, \tag{11}$$

where

$$\Pi(t) = \begin{bmatrix} I_S\otimes\mathcal{R} & \gamma\big((\mathsf{L}+\Phi)\otimes I_n\tilde{\mathcal{J}}\big)^\top\\ \gamma\big((\mathsf{L}+\Phi)\otimes I_n\tilde{\mathcal{J}}\big) & 0_{nS\times nS}\end{bmatrix},$$
$$\Psi(t) = \begin{bmatrix}-\mathbf{e}_S\otimes\mathbf{p}\\ -\omega(t)\end{bmatrix},\quad \begin{bmatrix}\mathbf{u}_{\dot{\vartheta}}(t)\\ \mathbf{u}_{\kappa}(t)\end{bmatrix} = \begin{bmatrix}\tilde{\dot{\vartheta}}(t)\\ \kappa(t)\end{bmatrix}.$$

Thus, the strategy update of the ith manipulator depends on

$$\Pi^i_{\dot{\vartheta}}(t)\mathbf{u}^i_{\dot{\vartheta}}(t) = \begin{cases}\Psi^i_{\dot{\vartheta}}(t), & \text{if } \mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) > 0,\\ \text{previous strategy}, & \text{if } \mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big)\le 0,\end{cases} \tag{12}$$

where $\mathbf{u}^i_{\dot{\vartheta}}(t) = \dot{\vartheta}_i(t) \in \mathbb{R}^m$; $\Pi^i_{\dot{\vartheta}}(t) = \{\Pi(t)\}_{r\times v}$ with $\{\cdot\}_{r\times v}$ representing extracting an r-by-v submatrix from $\{\cdot\}$, and $r = v \in [m(i-1)+1, mi]$; $\Psi^i_{\dot{\vartheta}}(t) = \{\Psi(t)\}_r$. The definition of $\mathbf{u}^i_{\kappa}(t) \in \mathbb{R}^n$ is similar to that of $\mathbf{u}^i_{\dot{\vartheta}}(t)$.

Remark 3: Define t_l and t_{l+1} as the lth and $l+1$th trigger instant of the manipulator i, respectively, and stipulate that before the new trigger instant (e.g., t_{l+1}) arrives, the strategy of manipulator is determined by the update at the previous instant (e.g., t_l). Furthermore, suppose that the strategy followed by the ith manipulator at $t \in [t_l, t_{l+1})$ is $\mathbf{u}^i_{\dot{\vartheta}}(t_l) = \dot{\vartheta}^\sharp_i$, so the joint velocity error accumulated by the manipulator in this time slice is $\varsigma_i(t) = \mathbf{u}^i_{\dot{\vartheta}}(t) - \dot{\vartheta}^\sharp_i$. Since $\dot{\vartheta}^\sharp_i$ is a time-independent vector, $|\mathrm{d}\varsigma_i/\mathrm{d}t| = |\dot{\mathbf{u}}^i_{\dot{\vartheta}}(t)|$ can be obtained for $\forall t \in [t_l, t_{l+1})$. Besides, considering the realizability of the task, the acceleration $\ddot{\mathbf{x}}_{\mathrm{d}}(t)$ of the end-effector is assumed to be bounded, so for the $\imath$th joint of the manipulator i, its acceleration $|\dot{\mathbf{u}}^i_{\dot{\vartheta}^\imath}(t)| = \ddot{\vartheta}^\imath_i(t)$ has a bound ℓ, i.e., $|\dot{\mathbf{u}}^i_{\dot{\vartheta}^\imath}(t)| < \ell$. Based on the above, the

velocity error of the ith joint at $t \to t_{l+1}$ is defined as $\lim_{t\to t_{l+1}}|\varsigma_i^i(t)| = \pounds > 0$, and then $t_{l+1} - t_l > \ell/\pounds > 0$ could be deduced according to the definition of $|\dot{\mathbf{u}}^i_{\dot{\vartheta}^i}(t)|$. Remarkably, there is a certain time interval between the lth and $l+1$th triggering of the manipulator, so the Zeno behavior will not occur. In another respect, the existing data acquisition modules generally take samples at a fixed sampling interval, and the manipulator only needs to perform a trigger judgment after a single data sampling. In this regard, there must be a minimum time interval between the two event triggers, so the Zeno behavior can also be excluded.

C. RNN for Real-Time Redundancy Resolution

In Section III-B, the strategy generation method for each manipulator has been specified. Moreover, in game theory, the combination of strategies that enables the manipulators to complete the cooperative task is known as Nash equilibrium.

The Nash equilibrium is expected to be found in the game of the manipulator. However, subject to many nonlinear factors in the multi-manipulator system, the direct calculation of (12) may be slow and belated (especially for $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) > 0$), which may impair the real-time performance of the manipulator to perform the task. Fortunately, neural networks provide an effective way to quickly solve complex problems. By decoupling a complex nonlinear problem into a linear problem from the error level, the calculation is simplified. Specifically, the deviation of the ith manipulator at the equilibrium point can be defined as

$$\varkappa^i_{\mathsf{u}}(t) = \Pi^i_{\mathsf{u}}(t)\mathbf{u}^i_{\mathsf{u}}(t) - \Psi^i_{\mathsf{u}}(t),$$

of which $\mathsf{u} \in \{\dot{\vartheta}, \kappa\}$. Then, to update the strategy in time, a recurrent neural network (RNN) solver based on deviation $\varkappa^i_{\mathsf{u}}(t)$ is constructed as

$$\dot{\varkappa}^i_{\mathsf{u}}(t) = -(\eta_1+\eta_2)\Gamma\big(\varkappa^i_{\mathsf{u}}(t)\big) - \eta_1\eta_2\int_0^t \Gamma\big(\varkappa^i_{\mathsf{u}}(\tau)\big)\mathrm{d}\tau, \quad (13)$$

where $\eta_1, \eta_2 > 0$ and $\Gamma(\cdot)$ is an activation function. Further, by expanding (13), one has

$$\begin{aligned}\Pi^i_{\mathsf{u}}(t)\dot{\mathbf{u}}^i_{\mathsf{u}}(t) = &-\dot{\Pi}^i_{\mathsf{u}}(t)\mathbf{u}^i_{\mathsf{u}}(t) - (\eta_1+\eta_2)\Gamma\big(\Pi^i_{\mathsf{u}}(t)\mathbf{u}^i_{\mathsf{u}}(t) - \Psi^i_{\mathsf{u}}(t)\big)\\ &- \eta_1\eta_2\int_0^t \Gamma\big(\Pi^i_{\mathsf{u}}(\tau)\mathbf{u}^i_{\mathsf{u}}(\tau) - \Psi^i_{\mathsf{u}}(\tau)\big)\mathrm{d}\tau + \dot{\Psi}^i_{\mathsf{u}}(t),\end{aligned} \quad (14)$$

which is the constructed solver for assisting the real-time strategy formulation of the manipulator. It is worth stressing that (14) starts a new round of calculations only if $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big) > 0$, and in other cases, it maintains the output of the previous round without providing updates to surrounding manipulators. With the aid of (14), the Nash equilibrium of the multi-manipulator game can be found readily.

Remark 4: Each startup of the RNN solver (14) means that the manipulator will abandon the previous strategy and generate a new strategy, which is the most computationally expensive part of the system. Besides, in order for each manipulator to acquire the behavior information of other manipulators in its communication area in real time, they need to interact frequently with surrounding manipulators. However, not every communication contains useful information, and not every updated strategy is significantly different from the previous one. Actually, at some time nodes, the manipulator can continue the previous strategy without having to send updates to surrounding manipulators. Note that the introduction of condition $\mathsf{D}_i\big(\mathfrak{L},\dot{\mathfrak{L}}\big)$ is like adding a switching mechanism to the system. When the stability condition is satisfied (i.e., $\mathsf{D}_i \le 0$), the ith manipulator only needs to maintain the previous strategy without starting the solver to perform computation and establishing communication with surrounding manipulators. This undoubtedly reduces the computing and communication burden of the system. A schematic block diagram is shown in Fig. 1 to illustrate the above process. It can be seen that each manipulator first goes through D_i to determine whether it is necessary to start the solver, and then performs the next operation and transmits the updated control signal to the multi-manipulator system. If the manipulator's error accumulates to the point where it could destabilize the system, D_i is activated to update the strategy.

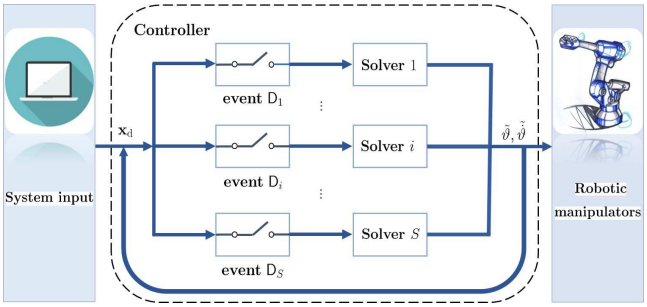


Fig. 1. Control block diagram of multi-manipulator system based on control strategy.

IV. Theoretical Analysis

In this section, the convergence of RNN solver (14) is analyzed. Besides, its performance under perturbation is studied.

Assumption 2: In the ensuing part, $\Gamma(\cdot)$ is set as a linear activation function. Notice that $\Gamma(\cdot)$ only affects the convergence rate of the solver without affecting its convergence.

A. Convergence Analysis

For RNN solver (14), it can be regarded as an interconnected system. Assuming that the strategy of the manipulator at the Nash equilibrium is $\mathbf{u}^{i*}_{\dot{\vartheta}^i}(t)$, the convergence of (14) can be judged by observing the trend of the deviation $\varkappa^i_{\mathsf{u}}(t)$.

Theorem 1: The strategy of the manipulator found by RNN solver (14) can converge globally to the equilibrium point of the game, at which point the deviation $\varkappa^i_{\mathsf{u}}(t)$ of the manipulator converges globally to zero.

Proof: The essence of RNN solver (14) is actually the expansion of design formula (13). Let $\Gamma(\mathsf{x}) = \mathsf{x}$, then (14) can be abbreviated to $\dot{\varkappa}^i_{\mathsf{u}}(t) = -(\eta_1+\eta_2)\varkappa^i_{\mathsf{u}}(t) - \eta_1\eta_2\int_0^t \varkappa^i_{\mathsf{u}}(\tau)\mathrm{d}\tau$ with $\varkappa^i_{\mathsf{u}}(t) = \Pi^i_{\mathsf{u}}(t)\mathbf{u}^i_{\mathsf{u}}(t) - \Psi^i_{\mathsf{u}}(t)$. Select a Lyapunov-candidate-function as

$$\mathbb{L}(t) = \big(\varkappa^i_{\mathsf{u}}(t)\big)^2 + \eta_1\eta_2\Big(\int_0^t \varkappa^i_{\mathsf{u}}(\tau)\mathrm{d}\tau\Big)^2.$$

Subsequently, finding the time derivative of $\mathbb{L}(t)$ yields

$$\dot{\mathbb{L}}(t) = 2\varkappa_{\mathsf{u}}^{i}(t)\dot{\varkappa}_{\mathsf{u}}^{i}(t) + 2\eta_1\eta_2\big(\int_0^t \varkappa_{\mathsf{u}}^{i}(\tau)\mathrm{d}\tau\big)\varkappa_{\mathsf{u}}^{i}(t). \quad (15)$$

By substituting (13) into (15), we can deduce that $\dot{\mathbb{L}}(t) = -2(\eta_1+\eta_2)(\varkappa_{\mathsf{u}}^{i}(t))^2$. Considering $\mathbb{L}(t) > 0$ and $\dot{\mathbb{L}}(t) < 0$, it can be concluded that RNN solver (14) is stable according to Lyapunov theory.

Let's prove it in another way. $\dot{\varkappa}_{\mathsf{u}}^{i}(t) = -(\eta_1+\eta_2)\varkappa_{\mathsf{u}}^{i}(t) - \eta_1\eta_2\int_0^t \varkappa_{\mathsf{u}}^{i}(\tau)\mathrm{d}\tau$ can be treated as a dynamic system in the form of an ordinary differential equation. By solving it directly, it is easy to infer that $\varkappa_{\mathsf{u}}^{i}(t)$ converges exponentially to zero no matter what initial state $\varkappa_{\mathsf{u}}^{i}(0)$ it starts from. The above two perspectives verify that the deviation of the manipulator at the equilibrium point converges globally to zero. Given that $\varkappa_{\mathsf{u}}^{i}(t) = \Pi_{\mathsf{u}}^{i}(t)\mathbf{u}_{\mathsf{u}}^{i*}(t) - \Psi_{\mathsf{u}}^{i}(t) = \mathbf{0}$ at the equilibrium point, $\mathbf{u}_{\mathsf{u}}^{i}(t)$ eventually converges to $\mathbf{u}_{\mathsf{u}}^{i*}(t)$. The proof is completed. ■

B. Robustness Analysis

Noise is a factor that cannot be ignored and affects the stability of the solver. A brief and slight perturbation may affect the calculation accuracy of the solver, resulting in jitter during the task execution. In reality, the errors in hardware implementation or external disturbances from environmental interference can be deemed as noise, where the truncation error and rounding error of a digital equipment pertain to the former, and the latter can be the electromagnetic noise, harmonic noise, shaking or others. If the fluctuation caused by noise cannot be eliminated in time, it will greatly reduce the performance of the actuator. Therefore, the perturbation inhibition ability is an objective requirement for the application of realistic scenarios.

Theorem 2: The RNN solver (14) polluted by bounded noise $\wp(t)$ stabilizes the driven system and further makes the steady-state error of its output bounded.

Proof: The linearly activated RNN solver (14) running in the bounded noise environment can be expressed as $\dot{\varkappa}_{\mathsf{u}}^{i}(t) = -(\eta_1+\eta_2)\varkappa_{\mathsf{u}}^{i}(t) - \eta_1\eta_2\int_0^t \varkappa_{\mathsf{u}}^{i}(\tau)\mathrm{d}\tau + \wp(t)$. Further, the $\imath$th subsystem of the ith manipulator can be abbreviated as $\dot{\mathcal{N}}_{\imath}(t) = \mathsf{G}\mathcal{N}_{\imath}(t) + \boldsymbol{w}\wp_{\imath}(t)$ with $\mathsf{G} = [-(\eta_1+\eta_2), -\eta_1\eta_2; 1, 0]$, $\mathcal{N}_{\imath}(t) = [\varkappa_{\mathsf{u}\imath}^{i}(t); \int_0^t \varkappa_{\mathsf{u}\imath}^{i}(\tau)\mathrm{d}\tau]$, and $\boldsymbol{w} = [1; 0]$, which is a typical linear system. It is evident to show that the eigenvalues of G are $\{-\eta_1, -\eta_2 < 0\}$, so G is Hurwitz matrix. It follows that RNN solver (14) is stable. Then, according to the basic result of BIBO stability, bounded input leads to bounded output. Therefore, for bounded noise $\wp(t)$ attached to the RNN solver (14), the fluctuation it brings is undoubtedly within a bounded range. The proof is completed. ■

Based on the conclusion of Theorem 2, the following theorems analyze the robustness of RNN solver (14) under unknown additive constant noise ß and unknown linear time-varying noise $\iota(t)$, respectively.

Theorem 3: Under the perturbation of unknown additive constant noise ß, the RNN solver (14) is able to find the strategy of global convergence to Nash equilibrium, at which point the deviation $\varkappa_{\mathsf{u}}^{i}(t)$ of the manipulator converges globally to zero.

Proof: First, the linearly activated RNN solver (14) running in the additive constant noise environment can be abbreviated as $\dot{\varkappa}_{\mathsf{u}}^{i}(t) = -(\eta_1+\eta_2)\varkappa_{\mathsf{u}}^{i}(t) - \eta_1\eta_2\int_0^t \varkappa_{\mathsf{u}}^{i}(\tau)\mathrm{d}\tau + ß$. Next, select the $\imath$th subsystem ($\imath \in \{1, 2, \ldots, m\}$) of the manipulator i to perform Laplace transform, and one has

$$s\varkappa_{\mathsf{u}\imath}^{i}(s) - \varkappa_{\mathsf{u}\imath}^{i}(0) = -(\eta_1+\eta_2)\varkappa_{\mathsf{u}\imath}^{i}(s) - \frac{\eta_1\eta_2}{s}\varkappa_{\mathsf{u}\imath}^{i}(s) + ß. \quad (16)$$

By calculation, the transfer function of (16) is $\mathsf{G}(s) = s/(s^2 + \eta_1 s + \eta_2 s + \eta_1\eta_2)$. Evidently, the poles of $\mathsf{G}(s)$ are all located on the left half-plane of the s-plane, so the system is stable. Further, utilizing the final value theorem leads to

$$\begin{aligned}\lim_{t\to\infty} \varkappa_{\mathsf{u}\imath}^{i}(t) &= \lim_{s\to 0} s\varkappa_{\mathsf{u}\imath}^{i}(s) \\ &= \lim_{s\to 0} \frac{s^2\big(\varkappa_{\mathsf{u}\imath}^{i}(0) + ß\big)}{s^2 + (\eta_1+\eta_2)s + \eta_1\eta_2} = 0.\end{aligned}$$

To sum up, $\varkappa_{\mathsf{u}\imath}^{i}(t)$ will converge to zero with time, which means $\mathbf{u}_{\mathsf{u}\imath}^{i}(t)$ will converge to $\mathbf{u}_{\mathsf{u}\imath}^{i*}(t)$ in the end. Considering that each subsystem can be proved by the above-mentioned process, it can be concluded that the RNN solver (14) is able to find the strategy to converge to Nash equilibrium even under the perturbation of unknown additive constant noise ß. The proof is completed. ■

Theorem 4: Under the perturbation of unknown linear time-varying noise $\iota(t) = \varrho t$, the deviation $\varkappa_{\mathsf{u}}^{i}(t)$ of the manipulator finally converges to $\|\varrho\|_2/(\eta_1\eta_2)$. Besides, the RNN solver (14) can achieve arbitrary output accuracy by adjusting the values of η_1 and η_2.

Proof: First, the linearly activated RNN solver (14) running in the unknown linear time-varying noise environment can be abbreviated as $\dot{\varkappa}_{\mathsf{u}}^{i}(t) = -(\eta_1+\eta_2)\varkappa_{\mathsf{u}}^{i}(t) - \eta_1\eta_2\int_0^t \varkappa_{\mathsf{u}}^{i}(\tau)\mathrm{d}\tau + \iota(t)$ with $\iota(t) = \varrho t$. Next, select the $\imath$th subsystem ($\imath \in \{1, 2, \ldots, m\}$) of the manipulator i to perform Laplace transform, and one has

$$s\varkappa_{\mathsf{u}\imath}^{i}(s) - \varkappa_{\mathsf{u}\imath}^{i}(0) = -(\eta_1+\eta_2)\varkappa_{\mathsf{u}\imath}^{i}(s) - \frac{\eta_1\eta_2}{s}\varkappa_{\mathsf{u}\imath}^{i}(s) + \frac{\varrho_{\imath}}{s^2}.$$

Further, utilizing the final value theorem leads to

$$\begin{aligned}\lim_{t\to\infty} \varkappa_{\mathsf{u}\imath}^{i}(t) &= \lim_{s\to 0} s\varkappa_{\mathsf{u}\imath}^{i}(s) \\ &= \lim_{s\to 0} \frac{s^2\big(\varkappa_{\mathsf{u}\imath}^{i}(0) + \varrho_{\imath}/s^2\big)}{s^2 + (\eta_1+\eta_2)s + \eta_1\eta_2} = \frac{\varrho_{\imath}}{\eta_1\eta_2}.\end{aligned}$$

Consequently, it can be easily obtained that $\varkappa_{\mathsf{u}}^{i}(t)$ eventually converges to $\|\varrho\|_2/\eta_1\eta_2$. Besides, notice that as long as $\eta_1\eta_2$ is large enough, one has $\varkappa_{\mathsf{u}}^{i}(t) \to \mathbf{0}$. Therefore, the output accuracy of the RNN solver (14) can be changed by adjusting the values of η_1 and η_2. The proof is completed. ■

The above theorems prove that even under noise interference, the RNN solver (14) is still able to find the strategy that converges to the Nash equilibrium with a steady-state error of zero or arbitrarily small, which is beneficial to the application in noisy scenes. To sum up, the proof on stability and robustness of RNN solver (14) is completed.

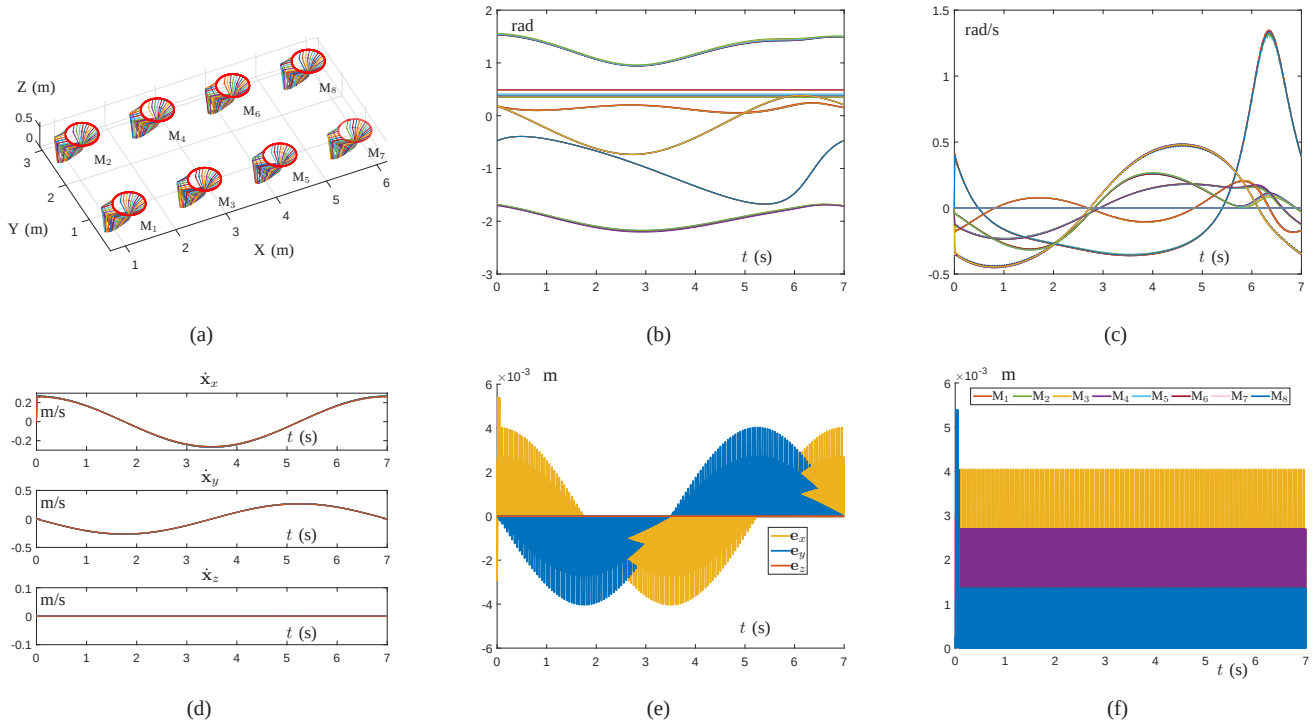


Fig. 2. Simulation results of a multi-manipulator system cooperatively tracking the circular trajectory using control strategy (12) under the perturbation of constant noise $\mathrm{Amp}(ß) = 10$. (a) Behavioral trajectories of manipulators performing tasks cooperatively. (b) Profiles of joint-angle. (c) Profiles of joint-velocity. (d) Profiles of end-effector speed. (e) Profiles of position tracking error in x-, y-, and z- directions. (f) Profiles of $\|\mathbf{e}_i\|_2$.

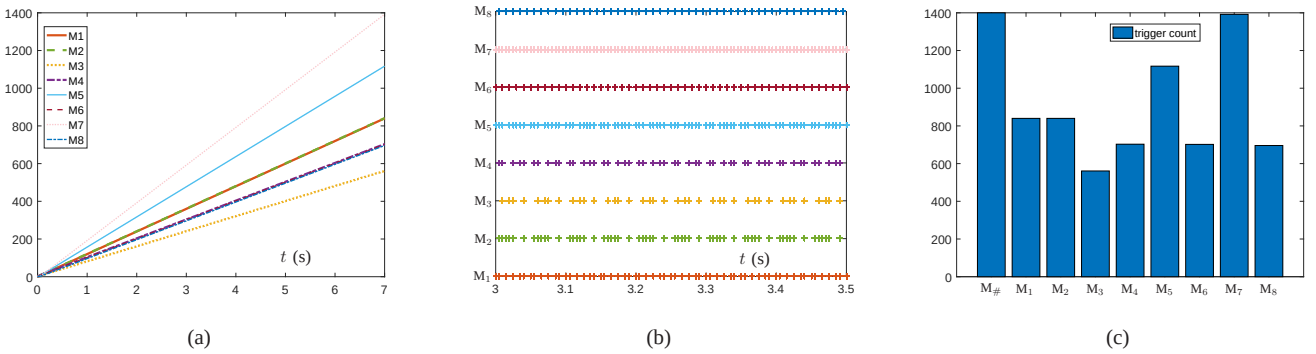


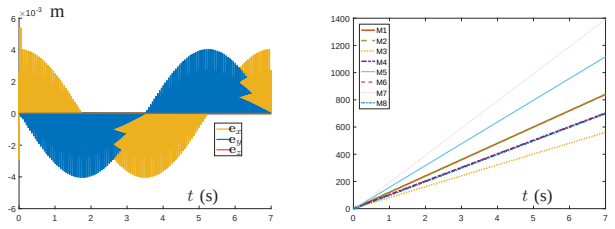
Fig. 3. Simulation results of controller update counts for the multi-manipulator system using control strategy (12) to execute the circular trajectory tracking task under the perturbation of constant noise $\mathrm{Amp}(ß) = 10$. (a) Control update counts. (b) Triggering time sequences (3 s to 3.5 s). (c) Total number of updates.

V. Simulation Results

In this section, a path-tracking task is assigned to manipulators to test the effectiveness of the proposed control strategy (12) and RNN solver (14). Furthermore, the control strategy is validated on the CoppeliaSim platform.

A. Circular Trajectory Tracking Task

In this subsection, eight PUMA 560 manipulators are configured to execute the circular trajectory tracking task, and their Denavit-Hartenberg (D-H) parameters are listed in Table I. The system uses a distributed network topology and full-duplex communication among adjacent manipulators (i.e., for $|i-j| = 1$, $\alpha_{ij} = 1$). Then, assigning command center sends instructions only to manipulators 1 and 5, and the remaining manipulators get information through neighboring manipulators (i.e., $\phi_1 = \phi_5 = 1$). Simulations are conducted on the MindSpore framework, and the specific parameters are set by considering the following requirements. Note that η_1 and η_2 are the design parameters of the solver (14), which can be used to adjust its convergence rate. Theoretically, larger values of η_1 and η_2 leads to a faster convergence rate of (14), which is revealed in Theorem 1. Therefore, we set $\gamma = \eta_1 = \eta_2 = 2000$ in the simulations. Additionally, λ is the design parameter of the event-triggered function. A larger value of λ leads to a more precise task execution, while the solver correspondingly consumes more resources. Therefore, after weighing the update cost and accuracy requirements, λ is set to 80 to complete the task with high accuracy while



(a) (b)

Fig. 4. Simulative experiment of the multi-manipulator system using control strategy (12) to track the circular trajectory under the perturbation of random noise $\mathrm{Amp}(ß) \in [9.8, 10.2]$. (a) Profiles of position tracking error in x-, y-, and z- directions. (b) Control update counts.

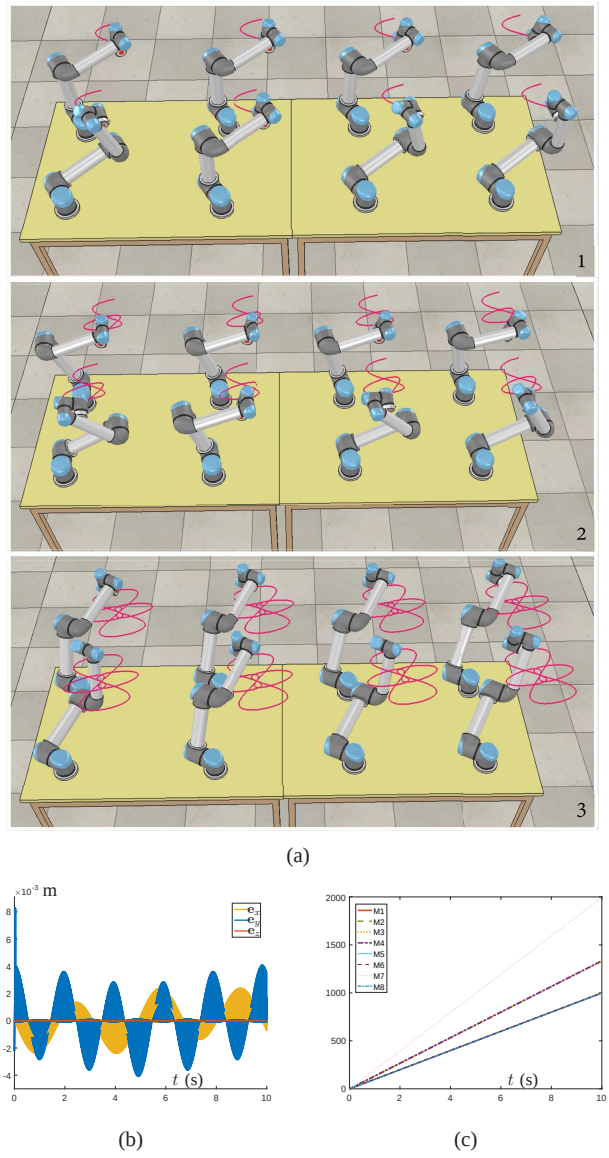
ensuring the reduction of communication overhead. Besides, $\mathcal{R} = I_m$ and $\mathbf{p} = \mathbf{0}$. In this setting, the payoff function is concretized into $\mathsf{P}_i(t) = -\dot{\vartheta}_i^\top(t)\dot{\vartheta}_i(t)/2$, which can be used to reduce the kinematic energy expenses for joint velocity $\dot{\vartheta}_i$. The execution period of this task is set to 7 s. Furthermore, suppose that the RNN solver (14) unfortunately suffers from the disturbance of constant noise ß whose amplitude is 10 (abbreviated as $\mathrm{Amp}(ß) = 10$) but the operation is not suspended.

In this scenario, the detailed simulation results are shown in Fig. 2. Concretely, Fig. 2(a) records the behavioral trajectory of manipulators cooperating to perform the task. It can be vividly seen that under the guidance of control strategy (12), the circular trajectory task assigned to the eight manipulators is smoothly completed. Next, Fig. 2(b) and Fig. 2(c) record the changes in joint angle and velocity of the manipulators during operation, separately, and Fig. 2(d) draws the end-effector velocity components in the x-, y-, and z- directions. Evidently, each end-effector achieves consensus in speed. To further illustrate the effect of the end-effector on the circular trajectory tracking task, the tracking error $\mathbf{e}_i = \mathbf{x}_i - \mathbf{x}_\mathrm{d}$ (components in the x-, y-, and z- directions) and its Euclidean norm $\|\mathbf{e}_i\|_2$ are given in Fig. 2(e) and (f), respectively. Especially in Fig. 2(f), the error $\|\mathbf{e}_i\|_2$ of each manipulator exhibits a fluctuating state. The reason for this phenomenon is that the trigger D_i will activate the RNN solver (14) to update the strategy only when it detects that the position error of the end-effector reaches a state that may cause the system to diverge.

TABLE I
D-H PARAMETERS OF PUMA 560

Link ι	$\alpha_{\iota-1}$ (rad)	$a_{\iota-1}$ (m)	d_ι (m)	Joint-angle ϑ^ι
1	0	0	0	ϑ^1
2	-π/2	0	0.15005	ϑ^2
3	0	0.43180	0	ϑ^3
4	-π/2	0.02030	0.43180	ϑ^4
5	π/2	0	0	ϑ^5
6	-π/2	0	0	ϑ^6

In this example, the sampling interval of the manipulator is set to 5×10^{-3} s. To test whether the manipulator with the control strategy (12) has the ability to reduce communication overhead, Fig. 3 statistics the update times of each manipulator during the task duration. Specifically, Fig. 3(a) shows the control update counts for each manipulator. Next, to display



(a)

(b) (c)

Fig. 5. Simulative experiment of UR5 robot system using control strategy (12) to track the bee curve under the perturbation of constant noise $\mathrm{Amp}(ß) = 10$. (a) Snapshots in task execution. (b) Profiles of position tracking error in x-, y-, and z- directions. (c) Control update counts.

the trigger status of manipulators intuitively, Fig. 3(b) shows the triggering time sequences of each manipulator in the period $t \in [3, 3.5]$ s. As seen in the results, the manipulator does not start the strategy update at every trigger instant, which undoubtedly reduces unnecessary communication and computing overhead. Subsequently, the total number of updates required for manipulators to complete the tracking task is counted in Fig. 3(c), where $\mathrm{M}_\#$ represents the manipulator that adopts the traditional periodic control techniques. By calculation, the manipulator with control strategy (12) updated 857 times on average, which reduces the total system overhead by 38.7% compared with the traditional periodic sampling technique. Additionally, in order to further verify the robustness of the proposed control strategy (12), Fig. 4 shows the simulation results in the presence of bounded random noise $\mathrm{Amp}(ß) \in [9.8, 10.2]$. The results show that the manipulator system can

TABLE II
COMPARISONS AMONG DIFFERENT CONTROL STRATEGIES FOR MANIPULATORS

	Number of manipulators	Network topology	Limited communication	Interference rejection	Communication overhead	Mathematical formulation	Optimality
This article	Multiple	Distributed	Y[a]	Y[b]	Y[c]	Game theory	Y[d]
Paper [2]	Single	N/A	N/A	N	N	Optimization	Y[d]
Paper [7]	two	Centralized	N	N	N	Optimization	Y[d]
Paper [8]	Multiple	Distributed	Y[a]	N	N	Game theory	Y[d]
Paper [9]	Multiple	Distributed	Y[a]	Y[b]	N	Game theory	Y[d]
Paper [11]	two	Decentralized	N	Y[b]	N	Adaptive control	Y[d]
Paper [15]	single	N/A	N/A	N	Y[c]	Optimization	Y[d]
paper [31]	Multiple	Distributed	Y[a]	Y[b]	N	Optimization	N
Paper [32]	Multiple	Decentralized	N	N	N	Optimization	Y[d]

[a] The method stipulates that the manipulator can only establish communication connection with adjacent manipulators.
[b] The method has the capability of suppressing noise.
[c] The method takes into account the communication overhead of the system.
[d] The method is able to find the optimal solution of the system globally.

still work as expected.

Furthermore, some comparisons of the different control methods of the manipulator are presented in Table II. It is worth pointing out that the existing methods with communication overhead considered are generally developed based on a single manipulator, e.g., [15], but are rarely promoted to multi-manipulator system. Moreover, the manipulators in [2], [7], [8], [32] are assumed to operate in a scene without noise, but the actual working environment may not be ideal. The method proposed in this article makes up for the above-mentioned deficiencies, and can still ensure the completion of tasks with lower communication overhead in a perturbed environment.

B. Application to UR5 Robot Control

To visually demonstrate the control effect of the control strategy (12) on the robot system, a group of UR5 robots are provided with experiments to perform the bee curve tracking task on CoppeliaSim. It is worth pointing out that CoppeliaSim integrates the physical model of the real robot, which can assist relevant practitioners to verify the designed control scheme. The setting of relevant parameters in this experiment is the same as that in Section V-A. Moreover, the task period of the task is set to 10 s. Snapshots of multiple UR5 robots tracking the bee curve in cooperation are shown in Fig. 5(a), which records the posture of several typical instants of the UR5 robot. Visibly, the task is successfully completed. Then, Fig. 5(b) records the position error components generated by each UR5 robot in the x-, y-, and z- directions during the task execution. Figure 5(c) sums the triggering times of each manipulator. Through calculation, the UR5 robot in the system under the guidance of control strategy (12) updates the strategy 1249 times on average, which is 37.5% lower than that of traditional time trigger. The above experiment shows that the proposed strategy (12) can also be well applied to the UR5 robot system. As a matter of fact, this study provides theoretical support for controlling multi-robot systems, especially for multi-robot systems with communication overhead considered.

VI. CONCLUSION

This article has studied a cooperative control scheme for multi-manipulator system with improved communication efficiency. Compared with the traditional periodic control, the proposed control strategy determines whether the controller needs to trigger an update by monitoring the measurement error of the manipulator. This measure has positive significance for reducing the network congestion of the manipulator system and improving communication efficiency. In addition, the cooperation and restriction relationship of manipulators in the system has been vividly reflected in the established game, and each manipulator is looking for its own optimal strategy that tends to the Nash equilibrium. Finally, examples have revealed that the proposed control strategy can effectively reduce the communication overhead while ensuring the cooperative performance of the manipulator. A possible future research direction is to develop a necessary system theory to quantitatively evaluate the control cost of the control strategy in calculation and communication, so as to further promote its application in practical scenarios.

REFERENCES

[1] S. Kartakis, A. Fu, M. Mazo, and J. A. McCann, "Communication schemes for centralized and decentralized event-triggered control systems," *IEEE Trans. Control Syst. Tech.*, vol. 26, no. 6, pp. 2035–2048, Nov. 2018.

[2] L. Jin, Z. Xie, M. Liu, C. Ke, C. Li, and C. Yang, "Novel joint-drift-free scheme at acceleration level for robotic redundancy resolution with tracking error theoretically eliminated," *IEEE/ASME Trans. Mechatron.*, In Press with DOI 10.1109/TMECH.2020.3001624.

[3] M. Hwang and D. Kwon, "Strong continuum manipulator for flexible endoscopic surgery," *IEEE/ASME Trans. Mechatron.*, vol. 24, no. 5, pp. 2193–2203, Oct. 2019.

[4] Y. Zhu, J. Qiao, and L. Guo, "Adaptive sliding mode disturbance observer-based composite control with prescribed performance of space manipulators for target capturing," *IEEE Trans. Ind. Electron.*, vol. 66, no. 3, pp. 1973–1983, Mar. 2019.

[5] S. Liu, D. Xing, Y. Li, J. Zhang, and D. Xu, "Robust insertion control for precision assembly with passive compliance combining vision and force information," *IEEE/ASME Trans. Mechatron.*, vol. 24, no. 5, pp. 1974–1985, Oct. 2019.

[6] Y. Wang, L. Cheng, Z. Hou, J. Yu, and M. Tan, "Optimal formation of multirobot systems based on a recurrent neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 322–333, Feb. 2016.

[7] L. Jin and Y. Zhang, "G2-type SRMPC scheme for synchronous manipulation of two redundant robot arms," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 153–164, Feb. 2015.

[8] S. Li, J. He, Y. Li, and M. U. Rafique, "Distributed recurrent neural networks for cooperative control of manipulators: A game-theoretic perspective," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 415–426, Feb. 2017.

[9] J. Zhang, L. Jin, and L. Cheng, "RNN for perturbed manipulability optimization of manipulators based on a distributed scheme: A game-theoretic perspective," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5116–5126, Dec. 2020.
[10] Y. Zhang and S. Li, "Adaptive near-optimal consensus of high-order nonlinear multi-agent systems with heterogeneity," *Automatica*, vol. 85, pp. 426–432, Nov. 2017.
[11] W. Gueaieb, F. Karray, and S. Al-Sharhan, "A robust hybrid intelligent position/force control scheme for cooperative manipulators," *IEEE/ASME Trans. Mechatron.*, vol. 12, no. 2, pp. 109–125, Apr. 2007.
[12] B. D. O. Anderson, C. Yu and J. M. Hendrickx, "Rigid graph control architectures for autonomous formations," *IEEE Control Syst. Mag.*, vol. 28, no. 6, pp. 48–63, Dec. 2008.
[13] Y. Zhang, S. Li, and L. Liao, "Consensus of high-order discrete-time multiagent systems with switching topology," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 2, pp. 721–730, Feb. 2021.
[14] D. Xie, S. Xu, Z. Li, and Y. Zou, "Event-triggered consensus control for second-order multi-agent systems," *IET Control Theory Appl.*, vol. 9, no. 5, pp. 667–680, 2015.
[15] Y. Zhang, H. Huang, S. Li, J. Li, and L. He, "Event-triggered zeroing dynamics for motion control of Stewart platform," *J. Frankl. Inst.*, vol. 357, no. 11, pp. 6453–6470, Jul. 2020.
[16] Z. Sun, Y. Xia, L. Dai, and P. Campoy, "Tracking of unicycle robots using event-based MPC with adaptive prediction horizon," *IEEE/ASME Trans. Mechatron.*, vol. 25, no. 2, pp. 739–749, Apr. 2020.
[17] M. Zhao, C. Peng, W. He, and Y. Song, "Event-triggered communication for leader-following consensus of second-order multiagent systems," *IEEE Trans. Cybern.*, vol. 48, no. 6, pp. 1888–1897, Jun. 2018.
[18] H. Meng, H. Zhang, Z. Wang, and G. Chen, "Event-triggered control for semiglobal robust consensus of a class of nonlinear uncertain multiagent systems," *IEEE Trans. Autom. Control*, vol. 65, no. 4, pp. 1683–1690, Apr. 2020.
[19] S. Yang, S. Jeon, and J. Choi, "Efficient sampling for rapid estimation of 3-D stiffness distribution via active tactile exploration," *IEEE/ASME Trans. Mechatron.*, vol. 25, no. 4, pp. 1729–1738, Aug. 2020.
[20] A. Selivanov and E. Fridman, "Event-triggered H_∞ control: A switching approach," *IEEE Trans. Autom. Control*, vol. 61, no. 10, pp. 3221–3226, Oct. 2016.
[21] J. Sun, J. Yang, and S. Li, "Reduced-order GPIO based dynamic event-triggered tracking control of a networked one-DOF link manipulator without velocity measurement," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 3, pp. 725–734, May 2020.
[22] X. Mi and S. Li, "Event-triggered MPC design for distributed systems with network communications," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 1, pp. 240–250, Jan. 2018.
[23] A. Amini, A. Asif, and A. Mohammadi, "Formation-containment control using dynamic event-triggering mechanism for multi-agent systems," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 5, pp. 1235–1248, Sep. 2020.
[24] C. Nowzari and J. Cortés, "Distributed event-triggered coordination for average consensus on weight-balanced digraphs," *Automatica*, vol. 68, pp. 237–244, 2016.
[25] T. Liu, M. Cao, C. D. Persis, and J. M. Hendrickx, "Distributed event-triggered control for asymptotic synchronization of dynamical networks," *Automatica*, vol. 86, pp. 199–204, 2017.
[26] T. Xu, Y. Hao, and Z. Duan, "Fully distributed containment control for multiple Euler-Lagrange systems over directed graphs: An event-triggered approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 6, pp. 2078–2090, Jun. 2020.
[27] C. Yang, G. Peng, Y. Li, R. Cui, L. Cheng, and Z. Li, "Neural networks enhanced adaptive admittance control of optimized robot-environment interaction," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2568–2579, Jul. 2019.
[28] L. Xiao, B. Liao, S. Li, Z. Zhang, L. Ding, and L. Jin, "Design and analysis of FTZNN applied to the real-time solution of a nonstationary Lyapunov equation and tracking control of a wheeled mobile manipulator," *IEEE Trans. Ind. Inf.*, vol. 14, no. 1, pp. 98–105, Jan. 2018.
[29] C. Hu, T. Ou, H. Chang, Y. Zhu, and L. Zhu, "Deep GRU neural network prediction and feedforward compensation for precision multiaxis motion control systems," *IEEE/ASME Trans. Mechatron.*, vol. 25, no. 3, pp. 1377–1388, Jun. 2020.
[30] Z. Zhang, T. Fu, Z. Yan, L. Jin, L. Xiao, Y. Sun, Z. Yu, and Y. Li, "A varying-parameter convergent-differential neural network for solving joint-angular-drift problems of redundant robot manipulators," *IEEE/ASME Trans. Mechatron.*, vol. 23, no. 2, pp. 679–689, Apr. 2018.
[31] L. Jin, J. Zhang, X. Luo, M. Liu, S. Li, L. Xiao, and Z. Yang, "Perturbed manipulability optimization in a distributed network of redundant robots," *IEEE Trans. Ind. Electron.*, In Press with DOI 10.1109/TIE.2020.3007099.
[32] S. Li, H. Cui, Y. Li, B. Liu, and Y. Lou, "Decentralized control of collaborative redundant manipulators with partial command coverage via locally connected recurrent neural networks," *Neural Comput. Appl.*, vol. 23, pp. 1051–1060, Sep. 2013.
[33] W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *Proc. 51st IEEE Conf. Decision and Control*, 2012, pp. 3270–3285.
[34] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

Jiazheng Zhang received the B.E. degree in communication engineering from Lanzhou University, Lanzhou, China, in 2019, and he is currently pursuing the M.E. degree in information and communication engineering with the School of Information Science and Engineering in Lanzhou University.

He currently conducts cooperative research with the Academy of Pleteau Science and Sustainability, Xining, China. His main research interests include neural networks and multi-robot coordination.

Long Jin (Member, IEEE) received the B.E. degree in automation and the Ph.D. degree in information and communication engineering from Sun Yat-sen University, Guangzhou, China, in 2011 and 2016, respectively. He received postdoctoral training at the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, from 2016 to 2017. He received the excellent doctoral dissertation award of Chinese Association for Artificial Intelligence (CAAI).

His current research interests include neural networks, robotics, optimization, and intelligent computing.

Chenguang Yang (Senior Member, IEEE) received the Ph.D. degree in control engineering from the National University of Singapore, Singapore, in 2010and postdoctoral training in human robotics from the Imperial College London, London, U.K. He has been awarded EU Marie Curie International Incoming Fellowship, UK EPSRC UKRI Innovation Fellowship, and the Best Paper Award of the IEEE Transactions on Robotics as well as over ten international conference Best Paper Awards. He is a Co-Chair of the Technical Committee on Bio-mechatronics and Bio-robotics Systems (B2S), IEEE Systems, Man, and Cybernetics Society, and a Co-Chair of the Technical Committee on Collaborative Automation for Flexible Manufacturing (CAFM), IEEE Robotics and Automation Society. He serves as Associate Editor of a number of IEEE Transactions and other international leading journals. His research interest lies in human robot interaction and intelligent system design.