Multi-Agent Cooperative Motion Planning Based on Alternating Direction Method of Multipliers

Ruishuang Chen, *Graduate Student Member, IEEE*, Zhihui Liang, Jie Cheng, Pengcheng You, *Member, IEEE*, and Zaiyue Yang, *Senior Member, IEEE*

***Abstract*—For robotics, multi-agent cooperative motion planning (MACMP) is an important but complex problem since it contains highly coupled dynamics and collision-free constraints. In this letter, a general parallel solution framework is proposed based on alternating direction method of multipliers (ADMM), which decouples the complicated coupling constraints and decomposes original problem into independent sub-problems for each agent, thereby making the iterative computation of ADMM and the solving of sub-problems parallel. The convergence of ADMM is analyzed, and the simulation results verify that proposed framework can better balance solution optimality and computational efficiency, compared with interior point method (IPM) and progressively constrained dynamic optimization algorithm (PCDO).**

***Index Terms*—Alternating direction method of multipliers, motion planning, multi-agent system, optimal control.**

I. Introduction

IN THE field of robotics, motion planning has always been a classic and important topic, while the main task of multi-agent cooperative motion planning (MACMP) is to find a collision-free trajectory connecting the starting point to end point for each agent [1]. Compared with single agent, MACMP problem will be more intractable, since it contains dynamic collision-free constraints that are highly coupled across agents [2]. Meanwhile, as the number of agents increases, the dimension of coupling constraints of MACMP problem will become higher, which makes the problem more difficult and time-consuming [3]. Obviously, the key point of solution is to reduce the dimension of constraints and handle the coupling of constraints across different agents [4]. Existing literature can mainly be classified into decentralized and centralized frameworks to deal with these difficulties [5].

The decentralized framework can also be called as the reaction-based framework, and the typical and well-known methods mainly include reciprocal velocity obstacle (RVO) [6] method and its revisions, such as optimal reciprocal collision-avoidance (ORCA) [7] method. The main concept of decentralized framework is to let each agent make decision independently to achieve collision-free planning through local observation information [8]. Since each agent only needs to solve its own problem, the computation time is short and this framework can realize online planning. However, as each agent cannot obtain global information, the decentralized framework often yields less optimal solution [8].

Compared with decentralized framework, the methods based on centralized framework can formulate MACMP as a unified optimal control problem (OCP) with state constraints, and generally have better optimality [9], but the problem will have higher computational complexity. From the perspective of optimization, MACMP is essentially a highly non-convex nonlinear programming (NLP) problem. In order to deal with complex constraints, several methods have been proposed to solve the sequential planning problem for each agent based on prioritization [10]. Prioritization-based methods decouple the dynamic collision-free constraints in time domain [11], but the solution space may become small for agents with low priority and deadlock may occur. In order to decouple constraints, [12] solves the problem based on an improved version of alternating direction method of multipliers (ADMM), but its sub-problem still contains dynamic collision-free constraint, and the control variables are only optimized in velocity space. Similarly, [13], [14], [15] implement problem decomposition based on ADMM, but they do not consider the time coupling of dynamics constraints, and there are still coupling constraints across different agents in sub-problems.

In addition, there are also many centralized methods utilizing other different strategies to solve MACMP problem. Reference [16] proposes the progressively constrained dynamic optimization (PCDO) algorithm which constructs intermediate problems by incrementally adding constraints to realize the solving of problem. Reference [17] presents a parabolic relaxation technique to convexify the constraints and analyzes the

Manuscript received 18 July 2023; revised 21 September 2023; accepted 9 October 2023. Date of publication 16 October 2023; date of current version 1 November 2023. This work was supported in part by the Shenzhen Key Laboratory of Control Theory and Intelligent Systems under Grant ZDSYS20220330161800001, and in part by the Science, Technology and Innovation Commission of Shenzhen Municipality under Grant 20200925174707002. Recommended by Senior Editor S. Dey. *(Corresponding author: Zaiyue Yang.)*

Ruishuang Chen, Zhihui Liang, Jie Cheng, and Zaiyue Yang are with the Shenzhen Key Laboratory of Control Theory and Intelligent Systems, School of System Design and Intelligent Manufacturing, and the Guangdong Provincial Key Laboratory of Human-Augmentation and Rehabilitation Robotics in Universities, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: chenrs2020@mail.sustech.edu.cn; liangzh2021@mail.sustech.edu.cn; chengj2021@mail.sustech.edu.cn; yangzy3@sustech.edu.cn).

Pengcheng You is with the Department of Industrial Engineering and Management, College of Engineering, Peking University, Beijing 100871, China (e-mail: pcyou@pku.edu.cn).

Digital Object Identifier 10.1109/LCSYS.2023.3324663

problem in a higher dimensional space. The above methods can effectively obtain better quality solutions under certain conditions, but they are essentially direct solving to whole problem, so that the computation time and resource usage may increase as the problem becomes more complicated [18].

In this letter, we propose a centralized optimization framework based on parallel sub-problem decomposition and computation. Based on ADMM [19], [20], [21], the MACMP problem will be decomposed into two types of sub-problems: sub-problem containing only dynamics constraints and sub-problem containing only collision-free constraints. For each type of sub-problem, it also can be split into smaller sub-problems which can be assigned to each agent individually. This can make the iterative computation of ADMM and solving of sub-problem for each agent in parallel. Since we utilize a centralized framework, each agent only needs to maintain connection with the master node (central controller or specific agent), and this connectivity is undirected.

The remainder of this letter is organized as follows. Section II formulates the MACMP problem and Section III illustrates the methodology. Section IV proofs the convergence of ADMM. Section V discusses the simulation results and conclusion is presented in Section VI.

II. Benchmark Problem Formulation

Typically, the MACMP problem for N_a agents can be formulated as:

$$\min_{\mathbf{s},t_f} J(\mathbf{s},t_f) = \mu_t \cdot t_f + \mu_i \cdot \sum_{i=1}^{N_a} J_i(\mathbf{s}_i) \tag{1}$$

$$\text{s.t.} \quad \mathcal{F}(\mathbf{s},t_f) = \mathbf{0},\ \ \mathcal{O}_i(\mathbf{s}) \ge 0,\ \ \mathbf{s} \in B,\ \ i = 1,\dots,N_a \tag{2}$$

where $\mathbf{s} \in \mathbb{R}^{n\cdot N_a} = [\mathbf{s}_1^\top,\dots,\mathbf{s}_i^\top,\dots\mathbf{s}_{N_a}^\top]^\top$ is the state variables vector (including input variables) of all N_a agents, $\mathbf{s}_i$ is the state variables vector of agent i, $t_f \in \mathbb{R}^+$ is the travel time, $J : \mathbb{R}^{n\cdot N_a} \times \mathbb{R}^+ \to \mathbb{R}$ is the objective function of whole problem, $J_i : \mathbb{R}^n \to \mathbb{R}$ is the objective function of agent i (generally quadratic), μ_t, μ_i are the weights, $\mathcal{F} : \mathbb{R}^{n\cdot N_a} \to \mathbb{R}^{n\cdot N_a} = [\mathcal{F}_1,\dots,\mathcal{F}_i,\dots\mathcal{F}_{N_a}]^\top$ is the dynamics equation matrix (dynamics constraint), $\mathcal{O}_i : \mathbb{R}^{n\cdot N_a} \to \mathbb{R}$ is the collision-free constraint between agent i and other agent, $B = \bigcup_{i=1}^{N_a} B_i$ is the box constraint including initial and final conditions. For simplicity, we define $\mathbf{d} = [\mathbf{s}^\top, t_f]^\top \in \mathbb{R}^{n\cdot N_a+1}$ to represent the decision variables of whole optimization problem. It is noted that (1) is a typical MACMP model and can be applicable to most relevant scenarios.

Obviously, problem (1) is a highly non-convex NLP problem and $\mathcal{F}, \mathcal{O}_i$ are the main coupling constraints. In order to handle coupling constraints, we utilize ADMM to achieve the decomposition and decoupling of original problem.

For ease of discussion and transformation, we introduce the auxiliary decision variables $\tilde{\mathbf{d}}, \hat{\mathbf{d}}, \bar{\mathbf{d}}$ to represent different constraints, and obtain the equivalent problem of (1) as

$$\min_{\tilde{\mathbf{d}},\hat{\mathbf{d}},\bar{\mathbf{d}}} J(\tilde{\mathbf{d}}) \tag{3}$$

$$\text{s.t.} \quad \mathcal{F}(\tilde{\mathbf{d}}) = \mathbf{0},\ \ \mathcal{O}_i(\hat{\mathbf{d}},\bar{\mathbf{d}}) \ge 0, \{\tilde{\mathbf{d}},\hat{\mathbf{d}},\bar{\mathbf{d}}\} \in B,$$
$$\tilde{\mathbf{d}} = \hat{\mathbf{d}} = \bar{\mathbf{d}},\ \ i = 1,\dots,N_a \tag{4}$$

For brevity without causing ambiguity, we directly use $\mathcal{F}, \mathcal{O}_i, B$ in (2) to describe the (augmented) constraints in (4). It is noted that $\hat{\mathbf{d}}$ only satisfies the constraint $\mathcal{F}$, and $\bar{\mathbf{d}}, \hat{\mathbf{d}}$ only satisfy the constraint $\mathcal{O}_i$, and $\bar{\mathbf{d}}$ is the copy of $\hat{\mathbf{d}}$. Meanwhile, $\mathcal{O}_i$ can be written as

$$\mathcal{O}_i(\hat{\mathbf{d}},\bar{\mathbf{d}}) = \mathcal{O}_i(\hat{\mathbf{s}}_i, \bar{\mathbf{d}} \setminus \bar{\mathbf{s}}_i) \tag{5}$$

where $\hat{\mathbf{s}}_i \in \hat{\mathbf{d}}, \bar{\mathbf{s}}_i \in \bar{\mathbf{d}}$, and $\bar{\mathbf{d}} \setminus \bar{\mathbf{s}}_i$ means to remove $\bar{\mathbf{s}}_i$ from $\bar{\mathbf{d}}$. Actually, the constraint $\mathcal{O}_i$ indicates that agent i should maintain a certain safe distance from other agents.

Remark 1: The introduction of copy variable $\bar{\mathbf{d}}$ will greatly reduce the coupling of collision-free constraints between agents. Section III-B will show that $\bar{\mathbf{d}}$ makes the original dynamic collision-free constraints transformed into static constraints at each time step.

Furthermore, in order to transform the problem (3) into the standard form of ADMM, the indicator function is introduced.

Definition 1: Assuming that E is a subset on $\mathbb{R}^n$, the indicator function on E is defined as the mapping $\mathcal{I}_E : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and

$$\mathcal{I}_E(e) = \begin{cases} 0 & \text{if } e \in E \\ +\infty & \text{if } e \notin E \end{cases} \tag{6}$$

Based on Definition 1, we can obtain the indicator function of decision variable $\tilde{\mathbf{d}}$ as

$$\mathcal{I}_{E_\mathcal{F}}(\tilde{\mathbf{d}}) = \begin{cases} 0 & \text{if } \tilde{\mathbf{d}} \in E_\mathcal{F} \\ +\infty & \text{if } \tilde{\mathbf{d}} \notin E_\mathcal{F} \end{cases} \tag{7}$$

where $E_\mathcal{F} = \{\tilde{\mathbf{d}}{:}\mathcal{F}(\tilde{\mathbf{d}}) = \mathbf{0}, \tilde{\mathbf{d}} \in B\}$.

Similarly, we can obtain the indicator function of $\hat{\mathbf{d}}, \bar{\mathbf{d}}$ as

$$\mathcal{I}_{E_{\mathcal{O}_i}}(\hat{\mathbf{d}},\bar{\mathbf{d}}) = \begin{cases} 0 & \text{if } (\hat{\mathbf{d}},\bar{\mathbf{d}}) \in E_{\mathcal{O}_i} \\ +\infty & \text{if } (\hat{\mathbf{d}},\bar{\mathbf{d}}) \notin E_{\mathcal{O}_i} \end{cases} \tag{8}$$

where $E_{\mathcal{O}_i} = \{(\hat{\mathbf{d}},\bar{\mathbf{d}}) : \mathcal{O}_i(\hat{\mathbf{d}},\bar{\mathbf{d}}) \ge 0, (\hat{\mathbf{d}},\bar{\mathbf{d}}) \in B\}$.

Based on indicator functions (7), (8), the constraints in (4) can be moved into the objective function (3), resulting in an equivalent optimization problem:

$$\min_{\tilde{\mathbf{d}},\hat{\mathbf{d}},\bar{\mathbf{d}},\check{\mathbf{d}}} \ J(\tilde{\mathbf{d}}) + \mathcal{I}_{E_\mathcal{F}}(\tilde{\mathbf{d}}) + \sum_{i=1}^{N_a} \mathcal{I}_{E_{\mathcal{O}_i}}(\hat{\mathbf{d}},\bar{\mathbf{d}}) \tag{9}$$

$$\text{s.t.} \quad \tilde{\mathbf{d}} = \hat{\mathbf{d}} = \bar{\mathbf{d}} = \check{\mathbf{d}} \tag{10}$$

where $\check{\mathbf{d}}$ is the common global variable. The introduction of $\check{\mathbf{d}}$ transforms the problem (3), (4) into a global consensus problem (9), (10), which conforms to the standard form of ADMM [19] and can be further decomposed and decoupled.

III. Problem Decomposition and Parallel Solving

According to the framework of ADMM, we can obtain the augmented Lagrangian function of (9) as

$$\begin{aligned}
&\mathcal{L}_\sigma(\tilde{\mathbf{d}},\hat{\mathbf{d}},\bar{\mathbf{d}},\check{\mathbf{d}},\tilde{\mathbf{w}},\hat{\mathbf{w}},\bar{\mathbf{w}}) \\
&= J(\tilde{\mathbf{d}}) + \mathcal{I}_{E_\mathcal{F}}(\tilde{\mathbf{d}}) + \sum_{i=1}^{N_a} \mathcal{I}_{E_{\mathcal{O}_i}}(\hat{\mathbf{d}},\bar{\mathbf{d}}) \\
&\quad + \tilde{\mathbf{w}}^\top(\tilde{\mathbf{d}} - \check{\mathbf{d}}) + \hat{\mathbf{w}}^\top(\hat{\mathbf{d}} - \check{\mathbf{d}}) + \bar{\mathbf{w}}^\top(\bar{\mathbf{d}} - \check{\mathbf{d}}) \\
&\quad + \frac{\sigma}{2} \cdot \left(\|\tilde{\mathbf{d}} - \check{\mathbf{d}}\|_2^2 + \|\hat{\mathbf{d}} - \check{\mathbf{d}}\|_2^2 + \|\bar{\mathbf{d}} - \check{\mathbf{d}}\|_2^2 \right) \\
&= J(\tilde{\mathbf{d}}) + \mathcal{I}_{E_\mathcal{F}}(\tilde{\mathbf{d}}) + \sum_{i=1}^{N_a} \mathcal{I}_{E_{\mathcal{O}_i}}(\hat{\mathbf{d}},\bar{\mathbf{d}}) + \frac{\sigma}{2} \left\| \tilde{\mathbf{d}} - \check{\mathbf{d}} + \sigma^{-1}\tilde{\mathbf{w}} \right\|_2^2
\end{aligned}$$

$$- \frac{1}{2\sigma}\|\tilde{\mathbf{w}}\|_2^2 + \frac{\sigma}{2}\left\|\hat{\mathbf{d}} - \check{\mathbf{d}} + \sigma^{-1}\hat{\mathbf{w}}\right\|_2^2 - \frac{1}{2\sigma}\|\hat{\mathbf{w}}\|_2^2 + \frac{\sigma}{2}\left\|\bar{\mathbf{d}} - \check{\mathbf{d}} + \sigma^{-1}\bar{\mathbf{w}}\right\|_2^2 - \frac{1}{2\sigma}\|\bar{\mathbf{w}}\|_2^2 \quad (11)$$

where $\tilde{\mathbf{w}}, \hat{\mathbf{w}}, \bar{\mathbf{w}} \in \mathbb{R}^{n \cdot N_v + 1}$ are the Lagrangian multiplier vectors and $\sigma \in \mathbb{R}^+$ is the penalty coefficient. Based on (11), the iterative sub-problems of ADMM can be obtained as

$$\tilde{\mathbf{d}}^{k+1} = \underset{\tilde{\mathbf{d}}}{\operatorname{argmin}}\ J(\tilde{\mathbf{d}}) + \mathcal{I}_{E_{\mathcal{F}}}(\tilde{\mathbf{d}}) + \frac{\sigma}{2}\left\|\tilde{\mathbf{d}} - \check{\mathbf{d}}^k + \sigma^{-1}\tilde{\mathbf{w}}^k\right\|_2^2 \quad (12)$$

$$\hat{\mathbf{d}}^{k+1} = \underset{\hat{\mathbf{d}}}{\operatorname{argmin}} \sum_{i=1}^{N_a} \mathcal{I}_{E_{\mathcal{O}_i}}(\hat{\mathbf{d}}, \bar{\mathbf{d}}^k) + \frac{\sigma}{2}\left\|\hat{\mathbf{d}} - \check{\mathbf{d}}^k + \sigma^{-1}\hat{\mathbf{w}}^k\right\|_2^2 \quad (13)$$

$$\bar{\mathbf{d}}^{k+1} = \underset{\bar{\mathbf{d}}}{\operatorname{argmin}} \sum_{i=1}^{N_a} \mathcal{I}_{E_{\mathcal{O}_i}}(\hat{\mathbf{d}}^{k+1}, \bar{\mathbf{d}}) + \frac{\sigma}{2}\left\|\bar{\mathbf{d}} - \check{\mathbf{d}}^k + \sigma^{-1}\bar{\mathbf{w}}^k\right\|_2^2 \quad (14)$$

$$\check{\mathbf{d}}^{k+1} = \frac{1}{3}\left(\tilde{\mathbf{d}}^{k+1} + \hat{\mathbf{d}}^{k+1} + \bar{\mathbf{d}}^{k+1} + \frac{1}{\sigma}\left(\tilde{\mathbf{w}}^k + \hat{\mathbf{w}}^k + \bar{\mathbf{w}}^k\right)\right) \quad (15)$$

$$\begin{aligned} \tilde{\mathbf{w}}^{k+1} &= \tilde{\mathbf{w}}^k + \sigma\left(\tilde{\mathbf{d}}^{k+1} - \check{\mathbf{d}}^{k+1}\right) \\ \hat{\mathbf{w}}^{k+1} &= \hat{\mathbf{w}}^k + \sigma\left(\hat{\mathbf{d}}^{k+1} - \check{\mathbf{d}}^{k+1}\right) \quad (16) \\ \bar{\mathbf{w}}^{k+1} &= \bar{\mathbf{w}}^k + \sigma\left(\bar{\mathbf{d}}^{k+1} - \check{\mathbf{d}}^{k+1}\right) \end{aligned}$$

where k represents the number of iteration. It is noted that first sub-problem (12) is uncoupled from other sub-problems (13), (14), thus they can be solved in parallel in the iterations of ADMM. Meanwhile, first sub-problem (12) only contains the dynamics constraints, while second and third sub-problems (13), (14) only contain the collision-free constraints. This indicates that two main types of complicated constraints in the problem are decoupled.

Furthermore, each sub-problem can be decomposed into smaller sub-problems which can also be solved in parallel. In the following context, we will discuss in detail how to solve each sub-problem respectively.

A. Solving First Sub-Problem

The first sub-problem (12) in ADMM can be rewritten as

$$\tilde{\mathbf{d}}^{k+1} = \underset{\tilde{\mathbf{d}}}{\operatorname{argmin}} \mu_t \cdot \tilde{t}_f + \mu_i \cdot \sum_{i=1}^{N_a} J_i(\tilde{\mathbf{s}}_i) + \frac{\sigma}{2}\left\|\tilde{\mathbf{d}} - \check{\mathbf{d}}^k + \sigma^{-1}\tilde{\mathbf{w}}^k\right\|_2^2 \quad (17)$$

$$\text{s.t.} \quad \mathcal{F}_i(\tilde{\mathbf{s}}_i, \tilde{t}_f) = \mathbf{0},\ \tilde{\mathbf{s}}_i \in B_i,\ i = 1, \ldots, N_a \quad (18)$$

where $\tilde{t}_f, \tilde{\mathbf{s}}_i \in \tilde{\mathbf{d}}$. For subsequent variables in $\tilde{\mathbf{d}}, \hat{\mathbf{d}}, \bar{\mathbf{d}}$, etc., we will utilize the symbolic expression similar to $\tilde{t}_f, \tilde{\mathbf{s}}_i$.

Although (17) only contains the dynamics constraints of agents and can be considered as a classical optimal control problem, the dimensionality of constraints is so high that it is still difficult to solve directly.

It can be found that for problem (17), the dynamics constraints of different agents are only coupled in time. To eliminate this coupling, we propose a 'First-Solve-Then-Regulate-Time' (FSTRT) solution strategy [22]. The key concept of FSTRT is to first solve the individual optimal control sub-problem of each agent i to obtain different travel time $\tilde{t}_f^i$, then find the maximal travel time $\tilde{t}_m = \max\{\tilde{t}_f^1, \ldots, \tilde{t}_f^{N_a}\}$, and finally re-solve each optimal control sub-problem with $\tilde{t}_m$ as the final condition of time.

Based on FSTRT, problem (17) can first be rewritten as N_a smaller optimal control sub-problems:

$$\{\tilde{\mathbf{s}}_i, \tilde{t}_f^i\} = \underset{\tilde{\mathbf{s}}_i, \tilde{t}_f^i}{\operatorname{argmin}}\ \mu_t \cdot \tilde{t}_f^i + \mu_i \cdot J_i(\tilde{\mathbf{s}}_i) + \frac{\sigma}{2}\left\|\left[\tilde{\mathbf{s}}_i^\top, \tilde{t}_f^i\right]^\top - \check{\mathbf{d}}_i^k + \sigma^{-1}\tilde{\mathbf{w}}_i^k\right\|_2^2 \quad (19)$$

$$\text{s.t.} \quad \mathcal{F}_i(\tilde{\mathbf{s}}_i, \tilde{t}_f^i) = \mathbf{0},\ \mathbf{s}_i \in B_i \quad (20)$$

where $\check{\mathbf{d}}_i = [\check{\mathbf{s}}_i^\top, \check{t}_f]^\top$, and $\tilde{\mathbf{w}}_i$ represents the corresponding weight vector. For each agent i, problem (19) is decoupled in time and can be solved in parallel.

After solving (19), we can obtain $\tilde{t}_m^{k+1}$ as the unified time final condition to regulate the travel time for each agent i. The problem (19) can be further rewritten as

$$\tilde{\mathbf{s}}_i^{k+1} = \underset{\tilde{\mathbf{s}}_i}{\operatorname{argmin}}\ \mu_i \cdot J_i(\tilde{\mathbf{s}}_i) + \frac{\sigma}{2}\left\|\tilde{\mathbf{s}}_i - \check{\mathbf{s}}_i^k + \sigma^{-1}\tilde{\mathbf{w}}_{\mathbf{s}_i}^k\right\|_2^2 \quad (21)$$

$$\text{s.t.} \quad \mathcal{F}_i(\tilde{\mathbf{s}}_i) = \mathbf{0},\ \mathbf{s}_i \in B_i,\ \tilde{t}_f^i = \tilde{t}_m^{k+1} \quad (22)$$

where $\tilde{\mathbf{w}}_{\mathbf{s}_i}^k$ is the weight vector of $\tilde{\mathbf{s}}_i$. Obviously, problem (21) can also be solved in parallel and its solution constitutes the solution of first sub-problem (12), i.e., $\tilde{\mathbf{d}}^{k+1} = [\tilde{\mathbf{s}}_1^{k+1\top}, \ldots, \tilde{\mathbf{s}}_{N_a}^{k+1\top}, \tilde{t}_m^{k+1}]^\top$. It should be emphasized that problems (19), (21) are simple classical optimal control problems for each agent, which can be solved efficiently.

Intuitively, FSTRT strategy makes the solution of first sub-problem (12) lose certain optimality, but it realizes parallel solution of each agent, which will significantly improve the solvability and efficiency of problem. In the testing of Case 1, compared with directly solving the optimal control problem, the optimality loss of FSTRT does not exceed 0.5%, but the computation time can be improved by at least 42%. Although we exchange solution efficiency with optimality, the loss of FSTRT will not be too large, or even negligible.

B. Solving Second and Third Sub-Problems

Essentially, second sub-problem (13) and third sub-problem (14) are similar. They both only contain collision-free constraints, and only the iteration variables for solving are different. For brevity, we only discuss how to solve second sub-problem (13) in this part.

Moving the indicator function $\mathcal{I}_{E_{\mathcal{O}_i}}$ to the constraints, problem (13) can be rewritten as:

$$\hat{\mathbf{d}}^{k+1} = \underset{\hat{\mathbf{d}}}{\operatorname{argmin}}\ \frac{\sigma}{2}\left\|\hat{\mathbf{d}} - \check{\mathbf{d}}^k + \sigma^{-1}\hat{\mathbf{w}}^k\right\|_2^2 \quad (23)$$

$$\text{s.t.} \quad \mathcal{O}_i(\hat{\mathbf{d}}, \bar{\mathbf{d}}^k) \geq 0,\ \hat{\mathbf{d}} \in B,\ i = 1, \ldots, N_a \quad (24)$$

It should be pointed out that since $\bar{\mathbf{d}}^k$ is a constant vector in constraint $\mathcal{O}_i$, the coupled dynamic collision-free constraints are transformed into static obstacle avoidance constraints in each time step, which eliminates the states coupling between agents. Therefore, the problem can be rewritten as N_a smaller independent sub-problems for each agent:

$$\hat{\mathbf{s}}_i^{k+1} = \underset{\hat{\mathbf{s}}_i}{\text{argmin}} \ \frac{\sigma}{2}\left\|\hat{\mathbf{s}}_i - \check{\mathbf{s}}_i^k + \sigma^{-1}\hat{\mathbf{w}}_{\mathbf{s}_i}^k\right\|_2^2 \tag{25}$$

$$\text{s.t.} \quad \mathcal{O}_i(\hat{\mathbf{s}}_i, \bar{\mathbf{d}}^k \setminus \bar{\mathbf{s}}_i^k) \geq 0, \ \hat{\mathbf{s}}_i \in B_i \tag{26}$$

where $\hat{\mathbf{w}}_{\mathbf{s}_i}^k$ is the weight vector of $\hat{\mathbf{s}}_i$.

Furthermore, since the constraint $\mathcal{O}_i$ is essentially only related to the coordinate $\mathbf{p}_i = [x_i, y_i]^\top \in \mathbf{s}_i$ of agent i, problem (25) can be further simplified as

$$\hat{\mathbf{p}}_i^{k+1} = \underset{\hat{\mathbf{p}}_i}{\text{argmin}} \ \frac{\sigma}{2}\left\|\hat{\mathbf{p}}_i - \check{\mathbf{p}}_i^k + \sigma^{-1}\hat{\mathbf{w}}_{\mathbf{p}_i}^k\right\|_2^2 \tag{27}$$

$$\text{s.t.} \quad \mathcal{O}_i(\hat{\mathbf{p}}_i, \bar{\mathbf{p}}^k \setminus \bar{\mathbf{p}}_i^k) \geq 0, \ \hat{\mathbf{p}}_i \in B_i \tag{28}$$

where $\mathbf{p} = [\mathbf{p}_1^\top, \ldots, \mathbf{p}_{N_a}^\top]^\top$, $\hat{\mathbf{w}}_{\mathbf{p}_i}$ is the weight vector of $\hat{\mathbf{p}}_i$, and decision variables that are not related to collision-free constraints can be directly obtained based on $\check{\mathbf{d}}^k$.

Similar to (19), (21), problem (27) for each agent i is uncoupled, indicating that it can also be solved in parallel. For third sub-problem (14), its solution method is the same as second sub-problem.

Obviously, above processing realizes the simplification of collision-free constraints and parallel solution of second and third sub-problems, which can improve the solvability and computational efficiency of problem.

Algorithm 1 Multi-Agent Cooperative Motion Planning

```
1:  Initialization: $\bar{\mathbf{d}}^0, \check{\mathbf{d}}^0, \tilde{\mathbf{w}}^0, \hat{\mathbf{w}}^0, \bar{\mathbf{w}}^0, \sigma, k \leftarrow 0$
2:  repeat
3:    do in parallel {Between Block 1 and Block 2}
4:      do in parallel                                      ⎫
5:        Parallel solution for problem (19)                ⎪
6:      end do                                              ⎪
7:      Obtain $\tilde{t}_m^{k+1}$                                  ⎬ Block 1
8:      do in parallel                                      ⎪
9:        Parallel solution for problem (21)                ⎪
10:     end do                                              ⎪
11:     Obtain $\tilde{\mathbf{d}}^{k+1}$                                  ⎭
12:     do in parallel                                      ⎫
13:       Parallel solution for problem (27)                ⎪
14:     end do                                              ⎪
15:     Obtain $\hat{\mathbf{d}}^{k+1}$                                  ⎬ Block 2
16:     do in parallel                                      ⎪
17:       Parallel solution for problem (14)                ⎪
18:     end do                                              ⎪
19:     Obtain $\bar{\mathbf{d}}^{k+1}$                                  ⎭
20:   end do
21:   Update $\check{\mathbf{d}}^{k+1}$ via (15)
22:   Update $\tilde{\mathbf{w}}^{k+1}, \hat{\mathbf{w}}^{k+1}, \bar{\mathbf{w}}^{k+1}$ via (16)
23:   $k \leftarrow k + 1$
24: until $\mathcal{O}_i(\tilde{\mathbf{d}}^{k-1}) \geq 0, i = 1, \ldots, N_a$
Output: The solution of problem (1): $\mathbf{d} = \tilde{\mathbf{d}}^{k-1}$
```

C. Algorithm Framework

Based on above discussions, we can obtain the solution process of ADMM for original problem (1) and proposed algorithm is summarized as Algorithm 1.

It should be emphasized that the convergence criterion of ADMM iteration is generally to determine whether the iteration error satisfies the error tolerance. However, in order to solve the problem efficiently, we set that when the solution of first sub-problem (12) satisfies the collision-free constraints, the ADMM iteration is terminated and this solution will be taken as the final solution. Although constraints verification may incur additional runtime for each iteration, it can obviously reduce the overall solution time.

This setting is reasonable and has no impact on the convergence analysis of ADMM iteration. Since the objective function of first sub-problem is consistent with original problem, the solution to optimal control problem that satisfies the collision-free constraints already has a high quality. It should be noted that since the original problem is a non-convex problem, we can only obtain the local optimal solution, while the solution of optimal control that satisfies the collision-free constraint is a local optimal solution.

IV. Convergence Analysis of ADMM

In this section, the convergence proof of ADMM iteration in Algorithm 1 is given. For ease of discussion, we rewrite the problem (9) as its equivalent form:

$$\min \quad \mathcal{J}(\mathbf{d}_p) + \mathcal{R}(\mathbf{d}_c) \tag{29}$$

$$\text{s.t.} \quad \mathbf{1} \cdot \mathbf{d}_p - \mathbf{1} \cdot \mathbf{d}_c = \mathbf{0} \tag{30}$$

where $\mathbf{d}_p = [\tilde{\mathbf{d}}^\top, \hat{\mathbf{d}}^\top, \bar{\mathbf{d}}^\top]^\top$, $\mathbf{d}_c = [\check{\mathbf{d}}^\top, \check{\mathbf{d}}^\top, \check{\mathbf{d}}^\top]^\top$, $\mathcal{J}(\mathbf{d}_p) = J(\tilde{\mathbf{d}}) + \mathcal{I}_{E_\mathcal{F}}(\tilde{\mathbf{d}}) + \sum_{i=1}^{N_a} \mathcal{I}_{E_{\mathcal{O}_i}}(\hat{\mathbf{d}}, \bar{\mathbf{d}})$, $\mathcal{R}(\mathbf{d}_c) \equiv 0$ is a constant value function, and $\mathbf{1}$ is an identity matrix.

Before proving convergence, we first give some necessary definitions, assumptions, and lemmas, which are mainly based on the theoretical framework in reference [23].

Assumption 1: The sets $E_\mathcal{F}$ and $E_{\mathcal{O}_i}$ are all bounded closed sets.

Assumption 1 is easily satisfied since most agent dynamics systems and geometrically feasible regions can be considered as bounded closed sets.

Definition 2: An extended valued function $Q : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is called coercive if $Q(\mathbf{q}) \to +\infty$ as $\|\mathbf{q}\| \to +\infty$.

Lemma 1: The objective function $\mathcal{J}(\mathbf{d}_p) + \mathcal{R}(\mathbf{d}_c)$ of (29) is coercive on feasible set $\mathcal{E} = \{(\mathbf{d}_p, \mathbf{d}_c) : \mathbf{1} \cdot \mathbf{d}_p - \mathbf{1} \cdot \mathbf{d}_c = \mathbf{0}\}$.

Proof: Since $\mathcal{J}(\mathbf{d}_p) + \mathcal{R}(\mathbf{d}_c)$ contains indicator functions and their corresponding sets are all bounded closed sets, $\mathcal{J}(\mathbf{d}_p) + \mathcal{R}(\mathbf{d}_c) \to +\infty$ when $\|(\mathbf{d}_p, \mathbf{d}_c)\| \to +\infty$. Then it follows that Lemma 1. ■

Lemma 2: For any fixed and feasible $\mathbf{d}_p$ (or $\mathbf{d}_c$), the optimization problem $\text{argmin}_{\mathbf{d}_c}\{\mathcal{J} : \mathbf{1} \cdot \mathbf{d}_c = \mathbf{d}_p\}$ (or $\text{argmin}_{\mathbf{d}_p}\{\mathcal{J} : \mathbf{1} \cdot \mathbf{d}_p = \mathbf{d}_c\}$) has a unique minimizer. Meanwhile, the map $H(\mathbf{d}_p) \triangleq \text{argmin}_{\mathbf{d}_c}\{\mathcal{J} : \mathbf{1} \cdot \mathbf{d}_c = \mathbf{d}_p\}$ (or $H(\mathbf{d}_c) \triangleq \text{argmin}_{\mathbf{d}_p}\{\mathcal{J} : \mathbf{1} \cdot \mathbf{d}_p = \mathbf{d}_c\}$) is a Lipschitz continuous map.

Proof: Lemma 2 is obviously established. Since $\mathbf{1}$ is a identity matrix, equation $\mathbf{1} \cdot \mathbf{d}_c = \mathbf{d}_p$ (or $\mathbf{1} \cdot \mathbf{d}_p = \mathbf{d}_c$) has unique solution and the map H will be a linear operator. ■

Lemma 3: The indicator functions $\mathcal{I}_{E_\mathcal{F}}$ and $\mathcal{I}_{E_{\mathcal{O}_i}}$ are lower semi-continuous on the sets $E_\mathcal{F}$ and $E_{\mathcal{O}_i}$ respectively.

Proof: Since the indicator function on bounded closed set is lower semi-continuous and $E_{\mathcal{F}}$ and $E_{\mathcal{O}_i}$ are all bounded closed set, it follows that Lemma 3. ■

Based on above assumptions and lemmas, problem (29) satisfies the convergence conditions in reference [23], and the proof is identical to [23] so that the convergence theorem of proposed algorithm can be obtained as follow.

Theorem 1: Assume that the objective function in (1) is Lipschitz differentiable, then for sufficiently large σ, the solution sequence $(\tilde{\mathbf{d}}^k, \hat{\mathbf{d}}^k, \bar{\mathbf{d}}^k, \check{\mathbf{d}}^k, \tilde{\mathbf{w}}^k, \hat{\mathbf{w}}^k, \bar{\mathbf{w}}^k)$ computed by Algorithm 1 has limit points and these limit points are stationary points of augmented Lagrangian function $\mathcal{L}_\sigma(\tilde{\mathbf{d}}, \hat{\mathbf{d}}, \bar{\mathbf{d}}, \check{\mathbf{d}}, \tilde{\mathbf{w}}, \hat{\mathbf{w}}, \bar{\mathbf{w}})$.

V. Simulation Results

In this section, simulation results are presented to verify the feasibility and effectiveness of proposed algorithm. It is noted that proposed algorithm is a relatively general framework and has no special restrictions on the dynamics model of agent.

For brevity without loss of generality, we set all agents to be the common circular mobile vehicles with same radius r for simulation. In practical applications, the time is discrete and the trajectory of each agent consists of a series of way-points. Therefore, the dynamics model of agent i is discrete and can be written as:

$$\begin{aligned} x_i(n+1) - x_i(n) &= \frac{t_f}{N-1} v_i(n) \cdot \cos\theta_i(n) \\ y_i(n+1) - y_i(n) &= \frac{t_f}{N-1} v_i(n) \cdot \sin\theta_i(n) \\ \theta_i(n+1) - \theta_i(n) &= \frac{t_f}{N-1} \omega_i(n) \end{aligned} \tag{31}$$

where $(x_i, y_i), \theta_i, v_i, \omega_i$ are the coordinates, orientation angle, velocity, angular velocity for the center of agent i respectively, t_f is the traveling time, N is the number of way-point, $n = 1, 2, \ldots, N-1$, and $i = 1, 2, \ldots, N_a$. Meanwhile, the decision variable $\mathbf{d}$ can be denoted as:

$$\mathbf{d} = \left[\mathbf{s}(1)^\top, \mathbf{s}(2)^\top, \ldots, \mathbf{s}(n)^\top, \ldots, \mathbf{s}(N)^\top, t_f\right]^\top \tag{32}$$

where $\mathbf{s}(n) = [\mathbf{s}_1(n)^\top, \ldots, \mathbf{s}_i(n)^\top, \ldots, \mathbf{s}_{N_a}(n)^\top]^\top$, and $\mathbf{s}_i(n) = [x_i(n), y_i(n), \theta_i(n), v_i(n), \omega_i(n)]^\top$.

As a result, the whole MACMP problem can be set as:

$$\min_{\mathbf{d}} J(\mathbf{d}) = \mu_t \cdot t_f + \mu_i \cdot \sum_{i=0}^{Na} \sum_{n=0}^{N-1} \|v_i(n+1) - v_i(n)\|_2^2$$
$$+ \ \mu_i \cdot \sum_{i=0}^{Na} \sum_{n=0}^{N-1} \|\omega_i(n+1) - \omega_i(n)\|_2^2 \tag{33}$$
$$\text{s.t.} \quad (31), \ \mathcal{O}_i(\mathbf{d}) \geq 0, \ \mathbf{d} \in B, \ i = 1, 2, \ldots, N_a \tag{34}$$

$\mathcal{O}_i(\mathbf{d})$ is the collision-free constraint of agent i and its specific form can be written as:

$$\|\mathbf{p}_i(n) - \mathbf{p}_j(n)\|_2 \geq 2r + l_s, \ j = 1, 2, \ldots, N_a, \neq i \tag{35}$$

where $\mathbf{p}_i$ denotes the center coordinate of agent i and l_s is safety margin. Therefore, problem (25) is actually a quadratically constrained quadratic program (QCQP) problem.

We will discuss the performance of proposed algorithm based on two cases. To illustrate the effectiveness of proposed algorithm, we utilize the interior point method (IPM) [24] implemented by IPOPT and PCDO [16] algorithm to achieve comparison with proposed algorithm in Case 1.

For each smaller sub-problem (19), (21), (27), we also utilize IPOPT to solve and the accuracy of IPOPT is uniformly set to 10^{-06}. The platforms and versions for modeling and computation are Microsoft Windows 10(Intel Core i5-10400F), MATLAB(2020b), AMPL(3.6.7) and IPOPT(3.12.13).

TABLE I
Simulation Parameters

Parameter Setting	N 80	$\{\tilde{\mathbf{w}}^0, \hat{\mathbf{w}}^0, \bar{\mathbf{w}}^0\}$ $\{\mathbf{1}, \mathbf{1}, \mathbf{1}\}$	$\{r, l_s\}$ $\{0.05, 0.01\}$
$\{\mu_t, \mu_i\}$ $\{1, 1\}$	$\{x_i(1), \theta_i(1)\}$ $\{0, 0\}$	$\{v_i(1), \omega_i(1)\}$ $\{0, 0\}$	$\{x_i(N), \theta_i(N)\}$ $\{2, 0\}$
θ $[-3.14, 3.14]$	v $[0, 0.2]$	ω $[-1, 1]$	t_f $[0, 50]$

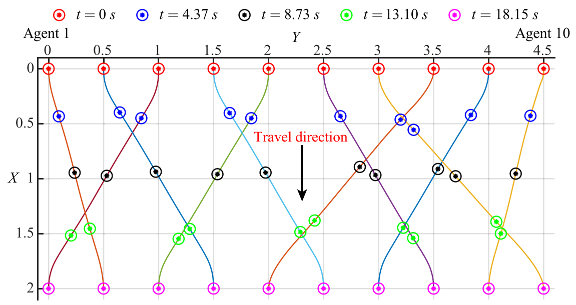

Fig. 1. Planned trajectories of proposed algorithm for 10 agents.

TABLE II
Comparison Results

Case study	Algorithm	Computation time	Travel time	Objective function value
5 Agents	IPM	25.95 s	12.91 s	13.10
	PCDO	18.56 s	12.92 s	13.09
	Proposed	9.03 s	17.30 s	17.83
10 Agents	IPM	Fail	-	-
	PCDO	68.67 s	13.10 s	13.33
	Proposed	19.47 s	18.15 s	19.47

A. Case 1: Cooperative Planning for 5 Agents and 10 Agents

In this case, we will verify the performance of proposed algorithm for 5 agents and 10 agents. The simulation parameters are listed in Table I, where the units are adopted as the International System of Units (SI). For 5 agents, the σ is set as 80 and for 10 agents, it is set as 1000.

In order to initialize $\bar{\mathbf{d}}^0, \check{\mathbf{d}}^0$ in Algorithm 1, we construct the initial position coordinates with a trivial straight line from start point to end point, while other variables are set to 0 and t_f is set as 1. It should be emphasized that above operation indicates that we do not provide an initial guess for proposed algorithm. Similarly, the initial guesses for IPM and PCDO are not provided.

The comparison results for 5 agents and 10 agents are listed in Table II, and the simulation results of proposed algorithm for 10 agents are depicted in Fig. 1, where the start and end

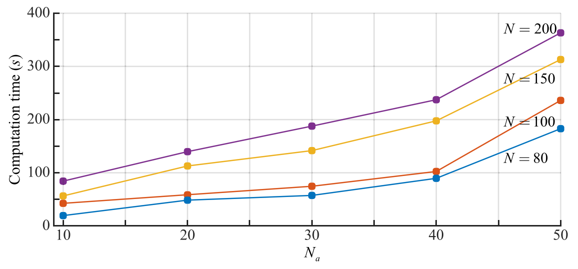

Fig. 2. Influence of the number of agent and waypoint.

points of each agent can be found. In Table II, proposed algorithm and PCDO can successfully solve the MACMP problem for 5 agents and 10 agents, but IPM only solves the problem for 5 agents and fails for 10 agents.

Although the optimality of IPM and PCDO is relatively better, the proposed algorithm has an obvious advantage in computation time and the calculated trajectories are still smooth, as shown in Fig. 1. Specially, for 10 agents, the objective function value of the proposed algorithm is 1.46 times that of PCDO, but the computation time of PCDO is 3.53 times that of the proposed algorithm. Therefore, the proposed algorithm can obviously improve the computational efficiency while guaranteeing a certain optimality.

B. Case 2: Influence of the Number of Agent and Waypoint

In actual application, the number of agent N_a and waypoint N both have the influence on computational efficiency of MACMP problem. Since the optimal control sub-problem in proposed framework is calculated in parallel, change in N_a mainly influences collision-free sub-problem, while change in N will influence each sub-problem. In this case, we will test the impact of different number of N_a and N on computational efficiency. The results are depicted in Fig. 2, which shows that increasing N_a and N will both increase the computation time. When N_a remains unchanged, although choosing a large N will make the trajectory smoother, it will increase the computation time, thus the selection of N needs to balance smoothness and solution efficiency [10].

VI. Conclusion

In this letter, we propose a relatively general algorithmic framework for solving MACMP problem. Based on ADMM, the original complex problem is decomposed into different types of sub-problems, while the introduction of copy variable and the proposal of FSTRT strategy further decouple sub-problems and realize parallel computing. The convergence of ADMM is analyzed, and the simulation and comparison results show that proposed algorithm can better balance optimality and computational efficiency.

References

[1] K. Solovey and D. Halperin, "On the hardness of unlabeled multi-robot motion planning," *Int. J. Robot. Res.*, vol. 35, no. 14, pp. 1750–1759, 2016.

[2] C. Tabasso, N. Mimmo, V. Cichella, and L. Marconi, "Optimal motion planning for localization of avalanche victims by multiple UAVs," *IEEE Control Syst. Lett.*, vol. 5, no. 6, pp. 2054–2059, Dec. 2021.

[3] L. Cohen, T. Uras, T. Kumar, and S. Koenig, "Optimal and bounded-suboptimal multi-agent motion planning," in *Proc. Int. Symp. Combinator. Search*, vol. 10, pp. 44–51.

[4] A. Britzelmeier, A. D. Marchi, and R. Richter, "Dynamic and nonlinear programming for trajectory planning," *IEEE Control Syst. Lett.*, vol. 7, pp. 2569–2574, 2023.

[5] F. Rossi, S. Bandyopadhyay, M. Wolf, and M. Pavone, "Review of multi-agent algorithms for collective behavior: A structural taxonomy," *IFAC-PapersOnLine*, vol. 51, no. 12, pp. 112–117, 2018.

[6] J. van den Berg, L. Ming, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2008, pp. 1928–1935.

[7] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proc. 14th Int. Symp. ISRR*, 2011, pp. 3–19.

[8] J. Alonso-Mora, P. Beardsley, and R. Siegwart, "Cooperative collision avoidance for nonholonomic robots," *IEEE Trans. Robot.*, vol. 34, no. 2, pp. 404–420, Apr. 2018.

[9] M. Leomanni, G. Mollica, A. Dionigi, P. Valigi, and G. Costante, "A convex programming approach to multipoint optimal motion planning for unicycle robots," *IEEE Control Syst. Lett.*, vol. 7, pp. 1688–1693, 2023.

[10] J. Li, M. Ran, and L. Xie, "Efficient trajectory planning for multiple non-holonomic mobile robots via prioritized trajectory optimization," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 405–412, Apr. 2021.

[11] D. R. Robinson, R. T. Mar, K. Estabridis, and G. Hewer, "An efficient algorithm for optimal trajectory generation for heterogeneous multi-agent systems in non-convex environments," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1215–1222, Apr. 2018.

[12] J. Bento, N. Derbinsky, J. Alonso-Mora, and J. S. Yedidia, "A message-passing algorithm for multi-agent trajectory planning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 521–529.

[13] R. V. Parys and G. Pipeleers, "Online distributed motion planning for multi-vehicle systems," in *Proc. Eur. Control Conf. (ECC)*, 2016, pp. 1580–1585.

[14] X. Zhang, Z. Cheng, J. Ma, L. Zhao, C. Xiang, and T. H. Lee, "Parallel collaborative motion planning with alternating direction method of multipliers," in *Proc. 47th Annu. Conf. IEEE Ind. Electron. Soc.*, 2021, pp. 1–6.

[15] A. Patnaik and A. R. Hota, "Optimization based collision avoidance for multi-agent dynamicalsystems in goal reaching task," 2021, *arXiv:2108.01320*.

[16] B. Li, Y. Zhang, Z. Shao, and N. Jia, "Simultaneous versus joint computing: A case study of multi-vehicle parking motion planning," *J. Comput. Sci.*, vol. 20, pp. 30–40, May 2017.

[17] C. Choi, M. Adil, A. Rahmani, and R. Madani, "Multi-robot motion planning via parabolic relaxation," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 6423–6430, Jul. 2022.

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

[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations Trends® Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[20] M. Doostmohammadian et al., "Distributed anytime-feasible resource allocation subject to heterogeneous time-varying delays," *IEEE Open J. Control Syst.*, vol. 1, pp. 255–267, 2022.

[21] M. Doostmohammadian, W. Jiang, and T. Charalambous, "DTAC-ADMM: Delay-tolerant augmented consensus ADMM-based algorithm for distributed resource allocation," in *Proc. IEEE 61st Conf. Decis. Control (CDC)*, 2022, pp. 308–315.

[22] B. Li, Y. M. Zhang, Y. M. Ge, Z. J. Shao, and P. Li, "Optimal control-based Online motion planning for cooperative lane changes of connected and automated vehicles," in *Proc. IEEE Int. Conf. Intell. Robots Syst. (IROS)*, 2017, pp. 3689–3694.

[23] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *J. Sci. Comput.*, vol. 78, no. 1, pp. 29–63, 2019.

[24] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.