Improved Hierarchical ADMM for Nonconvex Cooperative Distributed Model Predictive Control

Xiaoxue Zhang, Jun Ma, Zilong Cheng, Sunan Huang, Clarence W. de Silva, Fellow, IEEE, and Tong Heng Lee

Abstract—Distributed optimization is often widely attempted and innovated as an attractive and preferred methodology to solve large-scale problems effectively in a localized and coordinated manner. Thus, it is noteworthy that the methodology of distributed model predictive control (DMPC) has become a promising approach to achieve effective outcomes, e.g., in decision-making tasks for multi-agent systems. However, the typical deployment of such distributed MPC frameworks would lead to the involvement of nonlinear processes with a large number of nonconvex constraints. To address this important problem, the development and innovation of a hierarchical three-block alternating direction method of multipliers (ADMM) approach is presented in this work to solve this nonconvex cooperative DMPC problem in multi-agent systems. Here firstly, an additional slack variable is introduced to transform the original large-scale nonconvex optimization problem. Then, a hierarchical ADMM approach, which contains outer loop iteration by the augmented Lagrangian method (ALM) and inner loop iteration by three-block semi-proximal ADMM, is utilized to solve the resulting transformed nonconvex optimization problem. Additionally, it is analytically shown and established that the requisite desired stationary point exists for convergence in the algorithm. Finally, an approximate optimization stage with a barrier method is then applied to further significantly improve the computational efficiency, yielding the final improved hierarchical ADMM. The effectiveness of the proposed method in terms of attained performance and computational efficiency is demonstrated on a cooperative DMPC problem of decision-making process for multiple unmanned aerial vehicles (UAVs).

Index Terms—Alternating direction method of multipliers (ADMM), distributed optimization, model predictive control (MPC), multi-agent system, collision avoidance.

I. INTRODUCTION

With the rapid development of communication and computation technologies, cooperative decision making of large-scale multi-agent systems has become a promising trend in the development of automated systems [1]–[3]. Nevertheless, the significant challenge involved in cooperative decision making, in handling the coupling conditions among the interconnected agents with an appropriate guarantee of efficient computation still remains. In the existing literature, a significant amount of effort has been put into distributed control and coordinated networks for large-scale interconnected multi-agent systems, with the objective of optimizing the global cost while satisfying the required safety constraints collectively. Essentially for such approaches, the control and optimization algorithms deployed are distributed in structure, and the deployment utilizes only the local observations and information [4], [5]. With the appropriate communication technology in place, the agents are developed to suitably make decisions while attempting to satisfy all of the constraints or requirements invoked [6], [7]. On the other hand, a remarkable alternate development that utilizes an approach termed as distributed model predictive control (DMPC) has also been shown to display great promise to be effectively used to handle the input and state constraints with multiple objectives; and possibly also applicable to this important problem of cooperative decision making of large-scale multi-agent systems [8]. In addition, via the DMPC framework, there is great potential that the computational burden can be significantly relieved by decomposing the centralized system into several interconnected subsystems (with also a similar situation of availability of knowledge of necessary information among the connected agents).

For the DMPC problem with interconnected agents, the promising optimization approach of the alternating direction method of multipliers (ADMM) algorithm has the potential for very effective utilization. In this context, it can be noted that a large number of research works have demonstrated that the ADMM methodology has the capability to rather effectively determine the optimal solution to many challenging optimization problems, such as distributed optimization [9], decentralized control [10], [11], and statistical learning problems [12], [13]. The key idea of the ADMM approach is to utilize a decomposition coordination principle, wherein local sub-problems with smaller sizes are solved, and then the solutions of these subproblems can be coordinated to obtain the solution to the original problem. Notably, the ADMM solves the sub-problems by optimizing blocks of variables, such that efficient distributed computation can be attained. The convergence of the ADMM with two blocks has been investigated for convex optimization problems in [14], [15], and typical applications of such two-block ADMM approaches in distributed convex optimization problems are reported extensively in [16], [17]. Several researchers have also made rather noteworthy extensions from this two-block convex ADMM to multi-block convex ADMM in [18], [19]. However, although seemingly evident as a possibility with great potential, there has not been much substantial work....
on innovating the ADMM framework for nonconvex DMPC. Classical ADMM is known only to admit a linear convergence rate for convex optimization problems.

For nonconvex optimization problems, certain additional conditions and assumptions need to be considered to guarantee the required convergence \[20\]–\[26\]. More specifically, the work in \[20\] studies the two-block ADMM with one of the blocks defined as the identity matrix; and it also states that the objective function and the domain of the optimization variable need to be lower semi-continuous. Yet further, a nonconvex Bregman-ADMM is proposed in \[21\] by introducing a Bregman divergence term to the augmented Lagrangian function to promote the descent of certain potential functions. Moreover, the work in \[22\] proposes two proximal-type variants of ADMM with $\epsilon$-stationarity conditions to solve the nonconvex optimization problem with affine coupling constraints under the assumption that the gradient of the objective function is Lipschitz continuous. Additionally, in \[23\], a multi-block ADMM is presented to minimize a nonconvex and nonsmooth objective function subject to certain coupling linear equality constraints; and the work in \[24\] presents an ADMM-type algorithm for nonconvex optimization problem in the application environment of modern machine learning problems, though requiring the assumption that the penalty parameter should be large enough. Besides, an augmented Lagrangian based alternating direction inexact newton (ALADIN) method is introduced in \[25\] to solve the optimization problems with a separable nonconvex objective function and coupled affine constraints.

Noting all these additional conditions and assumptions that typically can arise, it is the situation where the ADMM approach cannot be directly applied to solve a large-scale constrained nonconvex program without further relaxation or reformulation. Also, the ADMM with more than two blocks is typically implemented with sequential optimization; and while a direct parallelization of a multi-block ADMM formulation is typically implemented with sequential optimization; and while the addition that it is typically less likely to converge \[28\], the aforementioned approach has the disadvantage that it is typically less likely to converge \[28\], \[29\]. In order to address all these limitations, a hierarchical three-block ADMM approach is utilized in our paper to solve the nonconvex optimization problem that arises for decision making in multi-agent systems. Firstly, an additional slack variable is innovatively introduced to transform the original large-scale nonconvex optimization problem, such that the intractable nonconvex coupling constraints are suitably related to the distributed agents. Then, the approach with a hierarchical ADMM that contains the outer loop iteration by the augmented Lagrangian method (ALM), and inner loop iteration by three-block semi-proximal ADMM, is utilized to solve the resulting transformed nonconvex optimization problem. Next, the approximate optimization with a barrier method is then applied to improve the computational efficiency. Finally, a multi-agent system involving decision making for multiple unmanned aerial vehicles (UAVs) is utilized to demonstrate the effectiveness of the proposed method in terms of attained performance and computational efficiency.

The remainder of this paper is organized as follows. Section \[II\] describes a multi-agent system that contains multiple interconnected agents equipped with sensors and communication devices, and also formulates a large-scale nonconvex optimization problem for decision making. Section \[III\] gives a more compact form of the optimization problem, which is further transformed by introducing slack variables. In section \[IV\] the hierarchical ADMM is used to solve the transformed optimization problem. Next, an improved version of such hierarchical ADMM is presented to accelerate the computation process based on the barrier method in Section \[V\]. Then, in Section \[VI\] a multi-UAV system is used as an example to show the effectiveness and computational efficiency of the hierarchical ADMM and improved hierarchical ADMM. Finally, pertinent conclusions of this work are given in Section \[VII\].

II. PROBLEM FORMULATION

The following notations are used in the remaining content. $\mathbb{R}^{a \times b}$ denotes the set of real matrices with $a$ rows and $b$ columns, $\mathbb{R}^a$ means the set of $a$-dimensional real column vectors. The symbol $X > 0$ and $X \geq 0$ mean that the matrix $X$ is positive definite and positive semi-definite, respectively. $x > y$ and $x \geq y$ mean that vector $x$ is element-wisely greater and no less than the vector $y$. $A^T$ and $x^T$ denote the transpose of the matrix $A$ and vector $x$. $I_t$ represents the $a$-dimensional identity matrix; $1_a$ and $1_{a,b}$ denote the $a$-dimensional all-one vector and the $a$-by-$b$ all-one matrix, respectively; $0_a$ and $0_{a,b}$ represent the $a$-dimensional all-zero vector and the $a$-by-$b$ all-zero matrix, respectively. The operator $\langle X, Y \rangle$ denotes the Frobenius inner product, i.e., $\langle X, Y \rangle = Tr(X^T Y)$ for all $X, Y \in \mathbb{R}^{a \times b}$; the operator $\| X \|$ is the Frobenius norm of matrix $X$. $\otimes$ denotes the Kronecker product and $\odot$ denotes the Hadamard product (Schur product). blockdiag($X_1, X_2, \ldots, X_n$) denotes a block diagonal matrix with diagonal entries $X_1, X_2, \ldots, X_n$;diag($a_1, a_2, \ldots, a_n$) denotes a diagonal matrix with diagonal entries $a_1, a_2, \ldots, a_n$, $Z^a$ and $Z^n_a$ mean the sets of positive integers $\{1, 2, \ldots, a\}$ and $\{a, a+1, \ldots, b\}$, respectively. $\Xi$ denotes the set of positive integers $\{1, 2, \ldots\}$. The operator $\{x_i\}_{i \in \Xi^n_a}$ means the concatenation of the vector $x_i$ for all $i \in \Xi^n_a$, i.e., $\{x_i\}_{i \in \Xi^n_a} = (x_a, x_{a+1}, \ldots, x_b) = [x_a^T \ x_{a+1}^T \ \ldots \ x_b^T]^T$.

A. Modelling of Interconnected Agents

In order to represent the constraint or information topology of multiple interconnected agents, an undirected graph $\mathcal{G}(V, E)$ can be utilized. The node set $V = \{1, 2, \ldots, N\}$ denotes the agents, and the edge set $E$ denotes the coupling constraints (information flow) between two interconnected agents, which is defined as

$$
\begin{array}{ll}
(i, j) \in E(t), & d_{safe} \leq \| p_i - p_j \| \leq d_{comm} \\
(i, j) \notin E(t), & \text{otherwise,}
\end{array}
$$

(1)

where $N$ and $M$ are the number of agents and coupling constraints in the multi-agent system, respectively, $p_i, p_j$ are the position vectors of the $i$th agent and $j$th agent, $d_{comm}$ and $d_{safe}$ mean the maximum communication distance and minimum safe distance between two agents, respectively. Therefore, based on the communication topology, an adjacency matrix of the network (denoted by $D$) can be obtained, which
is a square symmetric matrix. The elements of $\mathcal{D}$ indicate whether pairs of vertices are adjacent/connected or not in the graph. Therefore, the neighbor nodes of the $i$th agent are the corresponding indexes of nonzero elements of the $i$th row in $\mathcal{D}$, which is denoted by $\nu(i) = \{j|(i,j) \in \mathcal{E}, \forall j \in \mathcal{V}\}$.

**B. Problem Description**

Each connected agent in the set $\mathcal{V}$ has its origin and target state, which means each agent needs to achieve its task by coordinating with other connected agents in $\mathcal{V}$. Moreover, the information of the connected agents (neighbors) is necessary for each agent to make decisions, due to the requirement of the communication topology $\mathcal{G}$. Since the communication between connected agents occurs during the whole process, the present state information of neighbors needs to be conveyed to the $i$th agent to make decisions. Thus, the cooperative task can be formulated as a cooperative DMPC problem for each agent $i \in \mathcal{V}$. Then, given the initial state $x_i(t)$, the DMPC problem for the $i$th agent at timestamp $t$ can be written as

$$\begin{align*}
\min_{u_i, z_i} & \sum_{i \in \mathcal{V}} J_i(x_i, u_i) \\
\text{s. t.} & \quad x_i(t + 1) = A_i x_i(t) + B_i u_i(t) \quad \text{(2a)} \\
& \quad u_i(t) \leq u_i(t) \leq \overline{u}_i \quad \text{(2b)} \\
& \quad \underline{x}_i \leq x_i(t + 1) \leq \overline{x}_i \quad \text{(2c)} \\
& \quad (x_i(t + 1), u_i(t + 1)) \in \mathcal{C} \quad \text{(2d)} \\
& \quad \forall j \in \nu(i), \forall \tau \in \mathbb{Z}_t^{t+1}, \quad \text{(2e)}
\end{align*}$$

where the subscript $(\cdot)_i$ means the corresponding variable of the $i$th agent, $x_i \in \mathbb{R}^n$ and $u_i \in \mathbb{R}^m$ denote the state and control input vectors of the $i$th agent, respectively. $A_i \in \mathbb{R}^{n \times n}, B_i \in \mathbb{R}^{n \times m}$ are the state matrix and control input matrix, respectively. $t$ denotes the initial time stamp and $T$ is the prediction horizon. Notice that (2a) is the objective function for the state and control variables; (2b) denotes the kinematics of the agents; (2c) and (2d) denote the bound constraints of state and control input vectors, $\overline{x}_i$ and $\underline{x}_i$ ($\overline{u}_i$ and $\underline{u}_i$) are the upper (lower) bound of state and control input vectors, respectively; (2e) denotes the coupling constraints of two interconnected agents (the $i$th agent and the $j$th agent with $j \in \nu(i)$) and $\mathcal{C}$ denotes the set of the coupling constraints. The cost function of the $i$th agent $J_i(x_i, u_i)$ is a summation of the stage cost term $\ell_i$ and the terminal cost term $\ell_{T,i}$, i.e.,

$$J_i(x_i, u_i) = \ell_i(x_i(t+1), \cdots, x_i(t+T-1), u_i(t), \cdots, u_i(t+T-1)) + \ell_{T,i}(x_i(t+T)), \quad \text{(3)}$$

where the initial state $x_i(t)$ is known. The stage cost function $\ell_i$ is defined as

$$\ell_i(x_i, u_i) = \sum_{\tau=t}^{t+T-1} \langle x_i(\tau) - x_{i, ref}(\tau), Q_i(x_i(\tau) - x_{i, ref}(\tau)) \rangle + \sum_{\tau=t}^{t+T-1} \langle u_i(\tau), R_i u_i(\tau) \rangle, \quad \text{(4)}$$

where $x_{i, ref}$ is the reference state the $i$th agent tries to follow, and $Q_i \in \mathbb{R}^{n \times n}$ and $R_i \in \mathbb{R}^{m \times m}$ are the diagonal weighting matrices for the state and control input vectors, respectively. The terminal cost function $\ell_{T,i}$ is

$$\ell_{T,i}(x_i(t + T)) = \langle x_i(t + T) - x_{i, ref}(t + T), Q_{i,T}(x_i(t + T) - x_{i, ref}(t + T)) \rangle, \quad \text{(5)}$$

where $Q_{i,T}$ is the diagonal weighting matrix for the terminal state vector.

In a specific multi-agent system, $\mathcal{C}$ can be specified as

$$\mathcal{C} = \left\{ (x_i(\tau), x_j(\tau)) \mid d_{\text{safe}} \leq \|L_p(x_i(\tau) - x_j(\tau))\| \leq d_{\text{cmu}}, \forall i \in \mathcal{V}, \forall j \in \nu(i), \forall \tau \in \mathbb{Z}_t^{t+1} \right\}, \quad \text{(6)}$$

where $L_p$ is the locating matrix to extract the position vector $p_i$ from the state vector $x_i$, i.e., $p_i = L_p x_i(\tau)$.

**Assumption 1.** $\nu(i)$ in the prediction horizon $[t + 1, t + T]$ is time-invariant.

**III. Problem Transformation**

**A. Problem Reformulation**

First of all, the decision variable is set as

$$z_i = (u_i(t), x_i(t+1), u_i(t+1), x_i(t+2), \cdots, u_i(t + T - 1), x_i(t + T)) \in \mathbb{R}^{T(m+n)} \quad \text{(7)}$$

It is straightforward that $L_p x_i = p_i \in \mathbb{R}^{n_p}$, $\forall i \in \mathcal{V}$, where $n_p$ is the dimension of position in state vector $p_i$ and $n_p \leq n$, and $\|p_i(\tau) - p_j(\tau)\|^2 = (p_i(\tau) - p_j(\tau), p_i(\tau) - p_j(\tau))$. For all $j \in \nu(i)$ and $\tau \in \mathbb{Z}_t^{t+1}$, the coupling constraints $d_{\text{safe}} \leq |p_i(\tau) - p_j(\tau)| \leq d_{\text{cmu}}, \forall j \in \nu(i), \forall i \in \mathcal{V}, \forall \tau \in \mathbb{Z}_t^{t+1}$ can be equivalently expressed as

$$d_{\text{safe}}^2 \leq (p_i(t + 1) - p_j(t))^\top (p_i(t + 1) - p_j(t)) \leq d_{\text{cmu}}^2, \quad \forall j \in \nu(i),$$

$$d_{\text{safe}}^2 \leq (p_i(t + T) - p_j(t))^\top (p_i(t + T) - p_j(t)) \leq d_{\text{cmu}}^2, \quad \forall j \in \nu(i), \quad \text{(8)}$$

where $r_i$ is the number of the neighbors of the $i$th agent, i.e., $r_i = |\nu(i)|$ where $|\cdot|$ is the cardinality operator for the set $\nu(i)$. This means the $i$th agent has $r_i$ neighbors, and there are $r_i$ inequalities for each $\tau$. Thus, the coupling constraint $d_{\text{safe}} \leq |p_i(\tau) - p_j(\tau)| \leq d_{\text{cmu}}, \forall j \in \nu(i)$ can be equivalently expressed as

$$d_{\text{safe}}^2 \leq (p_i(\tau) - p_j(\tau))^\top (p_i(\tau) - p_j(\tau)) \leq d_{\text{cmu}}^2, \quad \forall j \in \nu(i), \forall \tau \in \mathbb{Z}_t^{t+1}.$$
\[ \begin{align*}
\left[ 0_{(n_v, n_p)} \quad I_{n_p} \quad 0_{(n_v, n_p-m)} \right] \quad \begin{bmatrix}
I_{n_v} \quad 0_{n_v, (n_p-n_v)} \quad I_{n_p}
\end{bmatrix}
\end{align*} \]

Besides, the mapping \( M : \mathbb{R} \rightarrow \mathbb{R} \) is given by
\[ M(z_i; p_j) = \left( I_{n_v} \otimes 1_{n_p}^T \right) \left( I_{n_v, n_p} \otimes \left( P(z_i; p_j) \right) 1_{n_p}^T \right). \]  

Therefore, the large-scale cooperative optimization problem (2) can be reformulated as:
\[ \begin{align*}
\min_{z_i} & \quad \sum_{i \in V} J_i(z_i) \\
\text{s.t.} & \quad G_i z_i = g_i \\
& \quad F_{z,i} z_i \leq f_{z,i} \\
& \quad M(z_i; p_j) P(z_i; p_j) \leq d_{\text{env}}^2 \\
& \quad M(z_i; p_j) P(z_i; p_j) \geq d_{\text{safe}}^2 \\
& \quad \forall j \in \nu(i), \forall i \in \mathbb{Z}_{i+1}^T,
\end{align*} \]

where \( J_i(z_i) = \langle z_i - z_{i, \text{ref}}, H_i (z_i - z_{i, \text{ref}}) \rangle \); \( H_i \) is the weighting matrix for the optimization variable \( z_i; \delta_{i, \text{ref}} \) is the reference vector, i.e., \( \delta_{i, \text{ref}} = 0_{m_v, x_{i, \text{ref}}(t + 1)}, 0_{m_v, x_{i, \text{ref}}(t + 2)}, \cdots, 0_{m_v, x_{i, \text{ref}}(t + T)} \). Due to the fact that the agents’ dynamic models are linear time-invariant during the optimization horizon \( [t, T] \), we can rewrite (2) as the equality constraint \( G_i z_i = g_i \) in (12), where
\[ G_i = I_T \otimes \left[ -B_i I_{n_i} \right] + \text{blockdiag} \left( 0_{(n_v, m)}, 0_{(n_v, m)} \right), \]
\[ g_i = \left[ \left( A_i x_i(t) \right)^T 0_n^T 0_n^T \cdots 0_n^T \right]^T + \left( -A_i 0_{(n_v, m)} \right) \left( -A_i 0_{(n_v, m, n)} \right) \]
and state vector \( x_i(t) \) at current timestamp \( t \). Notice that \( G_i \in \mathbb{R}^{T v \times T n (m+n)} \) and \( g_i \in \mathbb{R}^{T n} \). Additionally, (2d) and (2c) can be rewritten as \( F_{z,i} z_i \leq f_{z,i} \) in (12), where
\[ F_{z,i} = \begin{bmatrix}
I_{T(m+n)}^T & -I_{T(m+n)}^T
\end{bmatrix} \quad \text{and} \quad f_{z,i} = \begin{bmatrix}
1_T \otimes \left[ \eta_1^T \eta_2^T \cdots \eta_{n_p}^T \right]^T \\
1_T \otimes \left[ -\eta_1^T -\eta_2^T \cdots -\eta_{n_p}^T \right]^T
\end{bmatrix}.
\]

Remark 1. It is well-known that increasing the prediction horizon \( T \) can enlarge the domain of attraction of the MPC controller. However, this will increase the computational burden. Weighting the terminal cost can also enlarge the domain of attraction of the MPC controller without the occurrence of the terminal constraints; thus, the stabilizing weighting factor of a given initial state can be included, which has been proved in [30] that the asymptotic stability of the nonlinear MPC controller can be achieved.

By introducing an auxiliary variable \( z_{f,i} \in \mathbb{R}^{2T(m+n)} \), we have
\[ \begin{align*}
\min_{z_i} & \quad \sum_{i \in V} J_i(z_i) + \delta_{\text{env}}^2 (z_{f,i}) \\
\text{s.t.} & \quad A_i z_i + B_i z_{f,i} - h_i = 0 \\
& \quad M(z_i; p_j) P(z_i; p_j) \leq d_{\text{env}}^2 \\
& \quad M(z_i; p_j) P(z_i; p_j) \geq d_{\text{safe}}^2 \\
& \quad \forall j \in \nu(i),
\end{align*} \]

where \( A_i = \begin{bmatrix} G_i & F_{z,i} \end{bmatrix} \), \( B_i = \begin{bmatrix} 0_{Tn, 2T(m+n)} \end{bmatrix} \), and \( h_i = \begin{bmatrix} g_i & f_{z,i} \end{bmatrix} \). \( \delta_{\text{env}}^2 (z_{f,i}) \) is an indicator function defined as
\[ \delta_{\text{env}}^2 (z_{f,i}) = \begin{cases} 0, & \text{if } z_{f,i} \in \mathbb{R}_{+}^{2T(m+n)} \\
\infty, & \text{otherwise} \end{cases}. \]

Since the second inequality in (13) is nonlinear and nonconvex, we define a set
\[ Z_i = \left\{ z_i \in \mathbb{R}^{T(m+n)} \mid \delta_{Z_i}(z_i) \leq d_{\text{env}}^2 \right\}. \]

Then, the problem (13) can be simplified as
\[ \begin{align*}
\min_{z_i} & \quad J_i(z_i) + \delta_{Z_i}(z_i) + \delta_{\text{env}}^2 (z_{f,i}) \\
\text{s.t.} & \quad A_i z_i + B_i z_{f,i} - h_i = 0,
\end{align*} \]

where \( \delta_{Z_i}(z_i) \) is an indicator function of \( z_i \) onto the set \( Z_i \).

B. Optimization Problem Transformation

Due to the fact that the direct use of the conventional ADMM to the nonconvex optimization problem (16) directly cannot ensure the convergence, problem transformation is significant in the constrained nonconvex optimization problem (22). Thus, a slack variable \( s_i \in \mathbb{R}^{2T(m+n)} \) is introduced to the problem (16), and then the transformed optimization problem is given by
\[ \begin{align*}
\min_{z_i} & \quad J_i(z_i) + \delta_{Z_i}(z_i) + \delta_{\text{env}}^2 (z_{f,i}) + \delta_{\text{safe}}^2 (z_{f,i}) \\
\text{s.t.} & \quad A_i z_i + B_i z_{f,i} - h_i + s_i = 0,
\end{align*} \]

After introducing \( s_i \), the linear coupling constraints in (16) change from 2 blocks \( (z_i, z_{f,i}) \) to 3 blocks \( (z_i, z_{f,i} \text{ and } s_i) \). Since the block about \( s_i \) is an identity matrix \( I_{2T(m+n)} \) whose image is the whole space, there always exists an \( s_i \) such that \( A_i z_i + B_i z_{f,i} - h_i + s_i = 0 \) is satisfied, given any \( z_i \) and \( z_{f,i} \). In addition, the constraint \( s_i = 0 \) can be treated separately from the equality constraint \( A_i z_i + B_i z_{f,i} - h_i + s_i = 0 \). Therefore, we can consider the constraints in the problem (17) by separating them into two groups of constraints. The first group considers the equality constraint and the nonconvex constraint involved by the indicator function \( \delta_{Z_i}(z_i) \) and \( \delta_{\text{env}}^2 (z_{f,i}) \), and the other group deals with the constraint \( s_i = 0 \). As for the first sub-problem which does not include the constraint \( s_i = 0 \), some existing techniques of the ADMM can be employed. On the other hand, the constraint \( s_i = 0 \) can be transformed by dualizing the constant \( s_i = 0 \) with \( \lambda_i \) and adding a quadratic penalty \( \frac{\beta_i}{2} \| s_i \|^2 \). Then the problem (17) can be rewritten as
\[ \begin{align*}
\min_{z_i, s_{f,i}, s_i} & \quad J_i(z_i) + \delta_{Z_i}(z_i) + \delta_{\text{env}}^2 (z_{f,i}) \\
& \quad + \frac{\beta_i}{2} \| s_i \|^2 \\
\text{s.t.} & \quad A_i z_i + B_i z_{f,i} - h_i + s_i = 0,
\end{align*} \]

where \( \lambda_i \in \mathbb{R}_{2T(m+n)} \) is the dual parameter and \( \beta_i \in \mathbb{R}_{+} \) is the penalty parameter. Then, the ALM is used to solve the second sub-problem.
The augmented Lagrangian function of problem (18) is
\[ \mathcal{L}_{\rho_i} = J_i(z_i) + \delta_{\mathbb{R}_+^{2T+m+n}}(z_{f,i}) + \langle \lambda_i, s_i \rangle + \frac{\rho_i}{2} \| A_i z_i + B_i z_{f,i} - h_i + s_i \|^2, \]
where \( y_i \in \mathbb{R}^{2T+m+n} \) is the dual parameter of constraint \( A_i z_i + B_i z_{f,i} - h_i + s_i = 0 \) and \( \rho_i \in \mathbb{R} \) is the penalty parameter. Then, the first-order optimality conditions at a stationary solution \((z^*, z^*_{f,i}, s^*, y^*_{f,i})\) are given by
\[
\begin{align*}
0 &\in \nabla J_i(z^*_i) + \nabla Z_i(z^*_i) + A_i^T y^*_i, \quad \text{(20a)} \\
0 &\in B^T_i y^*_i + \nabla Z_i^{2T+m+n}(z^*_i), \quad \text{(20b)} \\
\lambda^*_i + \beta_i s^*_i + y^*_i &\in 0, \quad \text{(20c)} \\
A_i z^*_{i} + B_i z^*_{f,i} &- h_i + s^*_i = 0, \quad \text{(20d)}
\end{align*}
\]
where \( \nabla Z_i(z^*_i) \) and \( \nabla Z_i^{2T+m+n}(z^*_i) \) denote the normal cone of the set \( Z_i \) and \( \mathbb{R}_+^{2T+m+n} \) with respect to \( z^*_i \) and \( z^*_{f,i} \), respectively, and the superscript \((\cdot)^*\) means the variable in the \( k \)th iteration.

**Remark 2.** A solution that satisfies the optimality conditions (20) may not necessarily satisfy the primal feasibility of the constraint \( s_i = 0 \). However, ALM can prompt the slack variable \( s_i \) to zero through updating its dual parameter \( \lambda_i \).

Based on Remark 2, the update of \( \lambda_i \) helps to prompt \( s_i \) to zero with \( \lambda_i^{k+1} = \lambda_i^k + \beta_i s_i^k \).

**IV. Hierarchical ADMM**

For the problem (18), we treat it into two groups. The first one is to solve the problem (18) and update \( z_i, z_{f,i}, s_i, y_i \) in sequence with keeping \( \lambda_i \) and \( \beta_i \) constant, and the other group is to update the parameter \( \lambda_i \) and \( \beta_i \). In order to clarify the two groups, the iteration number of the first group (inner iteration) is indexed by \( r \), and the one of the second group (outer iteration) is \( k \).

1. **Update \( z_i \):**
\[
\begin{align*}
z_i^{k+1} &\in \arg \min_{z_i} \mathcal{L}_{\rho_i} \left( z_i, z_{f,i}^{k,r}, s_i^k, y_i^{k,r}, \lambda_i^k, \beta_i^k \right) \\
&= \arg \min_{z_i \in Z_i} \left( J_i(z_i) + \frac{\rho_i}{2} \left\| A_i z_i + B_i z_{f,i} \right\|^2 + s_i^k, y_i^{k,r} \right) - h_i + \frac{\| y_i^{k,r} \|}{\rho_i^k}.
\end{align*}
\]

2. **Update \( z_{f,i} \):**
\[
\begin{align*}
z_{f,i}^{k+1} &\in \arg \min_{z_{f,i}} \mathcal{L}_{\rho_i} \left( z_i, z_{f,i}^{k,r+1}, s_i^k, y_i^{k,r}, \lambda_i^k, \beta_i^k \right) \\
&= \arg \min_{z_{f,i}} \delta_{\mathbb{R}_+^{2T+m+n}}(z_{f,i}) + \frac{\rho_i}{2} \left\| A_i z_{i}^{k,r+1} + B_i z_{f,i} + s_i^k - h_i + \frac{\| y_i^{k,r} \|}{\rho_i^k} \right\|^2.
\end{align*}
\]

In order to obtain a close-form solution to the problem (22), we define \( T_k^r \in \mathbb{R}^{2T(m+n)} \) as a given positive semidefinite linear operator. Then, it follows that
\[
\begin{align*}
z_{f,i}^{k+1} &\in \arg \min \delta_{\mathbb{R}_+^{2T(m+n)}}(z_{f,i}) + \frac{1}{2} \left\| z_{f,i} - z_{f,i}^{k,r} \right\|^2_{T_k^r} + \frac{\rho_i}{2} \left\| A_i z_{i}^{k,r+1} + B_i z_{f,i} + s_i^k - h_i + \frac{\| y_i^{k,r} \|}{\rho_i^k} \right\|^2.
\end{align*}
\]

Then, the optimality condition to the sub-problem of \( z_{f,i} \) updating is given by
\[
\begin{align*}
0 &\in \delta_{\mathbb{R}_+^{2T(m+n)}}(z_{f,i}) + \rho_i B_i^T B_i z_{f,i} + T_i^r \left( z_{f,i} - z_{f,i}^{k,r} \right) + \rho_i^k B_i^T \left( A_i z_{i}^{k,r+1} + s_i^k - h_i + \frac{\| y_i^{k,r} \|}{\rho_i^k} \right).
\end{align*}
\]

Setting \( T_k^r = \alpha_i I_{2T(m+n)} - \rho_i B_i^T B_i \) where \( \alpha_i \) is the largest eigenvalue of the matrix \( B_i^T B_i \), we have
\[
\begin{align*}
0 &\in \left( \delta_{\mathbb{R}_+^{2T(m+n)}} + \alpha_i I_{2T(m+n)} \right) z_{f,i} - \alpha_i I - \rho_i^k B_i^T B_i \right) \left( z_{f,i} - z_{f,i}^{k,r} \right) + \rho_i^k B_i^T \left( A_i z_{i}^{k,r+1} + s_i^k - h_i + \frac{\| y_i^{k,r} \|}{\rho_i^k} \right).
\end{align*}
\]

Then, it follows that
\[
\begin{align*}
z_{f,i}^{k,r+1} &= \left( \alpha_i^{k+1} I_{2T(m+n)} - \rho_i B_i^T B_i \right)^{-1} \left( \alpha_i z_{f,i}^{k,r} \right) \\
&= \left( \alpha_i^{k+1} I_{2T(m+n)} - \rho_i^k B_i^T B_i \right)^{-1} \left( \alpha_i z_{f,i}^{k,r} \right) \\
&= \left( \alpha_i^{k+1} I_{2T(m+n)} - \rho_i^k B_i^T B_i \right)^{-1} \left( \alpha_i z_{f,i}^{k,r} \right).
\end{align*}
\]

**Lemma 1.** The projection operator \( \text{Proj}_C(\cdot) \) with respect to the convex set \( C \) can be expressed as
\[
\text{Proj}_C(\cdot) = (I + \alpha \partial_\delta C)^{-1},
\]
where \( \partial_\delta C \) is the sub-differential operator, and \( \alpha \in \mathbb{R} \) is an arbitrary number.

**Proof.** Define a finite dimensional Euclidean space \( X \) equipped with an inner product and its induced norm such that \( C \subset X \). For any \( x \in X \), there exists \( z \in X \) such that \( z \in (I + \alpha \partial_\delta C)^{-1}(x) \). Then, it follows that
\[
x \in (I + \alpha \partial_\delta C)(z) = z + \alpha \partial_\delta C.
\]

The projection operator \( \text{Proj}_C(\cdot) \) is
\[
\text{Proj}_C(z) = \arg \min_{x} \left\{ \partial_\delta C(x) + \frac{1}{2\alpha} \| z - x \|^2 \right\}.
\]

Due to the fact that the optimization problem (29) is strictly convex, the sufficient and necessary condition of (29) is
\[
0 \in \alpha \delta_C(z) + z - x.
\]

Since (30) is equivalent to (29) and the projection onto a convex set is unique, the operator \((I + \alpha \partial_\delta C)^{-1}\) is a point-to-point mapping. This completes the proof of Lemma 1.

\[\square\]
Based on Lemma 1, $z_{f,i}$ can be determined as

$$z_{f,i}^{k,r+1} = \text{Proj}_{\mathbb{R}^2}^{2T(m+n)} \left\{ \left( I_{2T(m+n)} - \alpha_i^{k-1} \rho_k^i B^T_i B_i \right) z_{f,i}^{k,r} + \alpha_i^{k-1} \rho_k^i B_i \left( A_i z_{i}^{k,r+1} + s_i^{k,r} - h_i + \frac{y_{i,r}^{k,r}}{\rho_k^i} \right) \right\},$$

where the projection operator $\text{Proj}_{\mathbb{R}^2}^{2T(m+n)}(z)$ means the projection of $z$ onto a closed convex set $\mathbb{R}^2_{+}^{2T(m+n)}$.

**Lemma 2.** With $C = \mathbb{R}^n$, then for any $z \in \mathbb{R}^n$, the projection of $z$ onto $C$ is given by

$$\text{Proj}_C(z) = \max\{z, 0\}.$$  \hfill (32)

According to Lemma 2, $z_{f,i}^{k,r+1}$ can be easily obtained.

**3. Update $s_i$:**

$$s_i^{k,r+1} = \arg \min_{s_i} \nabla \rho_k^i \left( z_{i}^{k,r+1}, z_{f,i}^{k,r+1}, s_i^{k,r}, \lambda_i^k, \rho_k^i, \beta_i^k \right)$$

$$= -\frac{\rho_k^i}{\rho_k^i + \beta_i^k} \left( A_i z_{i}^{k,r+1} + B_i z_{f,i}^{k,r+1} - h_i + \frac{y_{i,r}^{k,r}}{\rho_k^i} \right)$$

$$- \frac{\lambda_i^k}{\rho_k^i + \beta_i^k}.$$ \hfill (33)

**4. Update $y_i$:**

$$y_i^{k,r+1} = \arg \min_{y_i} \nabla \rho_k^i \left( z_{i}^{k,r+1}, z_{f,i}^{k,r+1}, s_i^{k,r+1}, y_i, \lambda_i^k, \rho_k^i, \beta_i^k \right)$$

$$= y_i^{k,r} + \rho_k^i \left( A_i z_{i}^{k,r+1} + B_i z_{f,i}^{k,r+1} + s_i^{k,r+1} - h_i \right).$$ \hfill (34)

**Assumption 2.** Given $\lambda_i^k, \beta_i^k$ and $\rho_k^i$, the $z_i$-update can find a stationary solution $z_i^{\ast}$ such that

$$0 \in \partial z_{i} \lambda_i^k (z_{i}^{\ast}, z_{f,i}^{\ast}, s_i^{\ast}, y_i^{\ast}),$$

$$\nabla \rho_k^i (z_{i}^{\ast}, z_{f,i}^{\ast}, s_i^{\ast}, y_i^{\ast}) \leq 0.$$ \hfill (35)

for all $a \in \mathbb{Z}_+$.  

**Lemma 3.** For all $a \in \mathbb{Z}_+$, the following conditions holds:

$$\langle B_i^T y_i^{\ast} - 1 \rangle + \rho_k^i B_i^T (A_i z_{i}^{\ast} + B_i z_{f,i}^{\ast} + s_i^{\ast} - h_i),$$

$$\langle z_{f,i}^{\ast} - z_{i}^{\ast} \rangle \geq 0, \forall a \in \mathbb{R}^{2T(m+n)}$$

$$\lambda_i^k + \beta_i^k s_i^{\ast} + y_i^{\ast} = 0.$$ \hfill (36)

**Proof.** This lemma can be straightforwardly derived based on the optimality conditions of $z_i^{\ast}$ and $z_{f,i}^{\ast}$-updates, i.e., (20a) and (20c), and thus the proof is omitted. \hfill \square

**Theorem 1.** With $\rho_k^i \geq \sqrt{2} \beta_i^k$, the inner iteration of problem (18) converges to the stationary point $(z_{i}^{k}, z_{f,i}^{k}, s_i^{k}, y_i^{k})$ of the transformed problem (18).

**Proof.** With Assumption 2, we have

$$\nabla \rho_k^i (z_{i}^{k,r}, z_{f,i}^{k,r}, s_i^{k,r}, y_i^{k,r}) \geq \nabla \rho_k^i (z_{i}^{k}, z_{f,i}^{k}, s_i^{k}, y_i^{k}).$$ \hfill (37)

Besides, we also have

$$\nabla \rho_k^i \left( z_{i}^{k,r}, z_{f,i}^{k,r}, s_i^{k,r}, y_i^{k,r} \right) - \nabla \rho_k^i \left( z_{i}^{k}, z_{f,i}^{k}, s_i^{k}, y_i^{k} \right)$$

$$= \langle B_i^T y_i^{k,r} - 1 \rangle + \rho_k^i B_i^T \left( A_i z_{i}^{k,r} + B_i z_{f,i}^{k,r} + s_i^{k,r} - h_i \right),$$

$$\langle z_{f,i}^{k,r} - z_{i}^{k,r} \rangle \geq \frac{\rho_k^i}{2} \left\| B_i (z_{i}^{k,r} - z_{f,i}^{k,r}) \right\|^2.$$ \hfill (38)

Note that the equality in (38) holds because of the following claim:

$$\left\| A_i z_{i}^{k} + s_i^{k-1} - h_i + B_i z_{f,i}^{k} \right\|^2 - \left\| A_i z_{i}^{k} - s_i^{k-1} - h_i + B_i z_{f,i}^{k} \right\|^2$$

$$= 2 \langle A_i z_{i}^{k} + s_i^{k-1} - h_i + B_i z_{f,i}^{k}, B_i z_{f,i}^{k-1} - B_i z_{f,i}^{k} \rangle + \left\| B_i z_{f,i}^{k-1} - B_i z_{f,i}^{k} \right\|^2.$$ \hfill (39)

In addition, the inequality in (38) can be obtained from Lemma 3. Now, the descent of $z_i$ and $z_{f,i}$ updates has been proved. Next, we will show the descent of $s_i$ and $y_i$ updates.

$$\nabla \rho_k^i \left( z_{i}^{k}, z_{f,i}^{k}, s_i^{k}, y_i^{k} \right) - \nabla \rho_k^i \left( z_{i}^{k-1}, z_{f,i}^{k-1}, s_i^{k}, y_i^{k} \right)$$

$$= \langle \alpha_i^k (s_i^{k-1} - s_i^{k}), (s_i^{k}) \rangle + \frac{\beta_k^i}{2} \left\| (s_i^{k}) - s_i^{k} \right\|^2$$

$$+ \langle y_i^{k}, s_i^{k} - s_i^{k} \rangle + \frac{\rho_k^i}{2} \left\| A_i z_{i}^{k} + B_i z_{f,i}^{k}, s_i^{k} - h_i \right\|^2$$

$$\geq \left( \frac{\rho_k^i}{2} - \frac{\beta_k^i}{2} \right) \left\| s_i^{k} - s_i^{k-1} \right\|^2.$$ \hfill (40)

Remarkably, the first equality holds due to (34) and the following derivation:

$$-\rho_k^i \langle A_i z_{i}^{k} + B_i z_{f,i}^{k} - h_i + s_i^{k}, A_i z_{i}^{k} + B_i z_{f,i}^{k} - h_i + s_i^{k-1} \rangle$$

$$+ \frac{\rho_k^i}{2} \left\| A_i z_{i}^{k} + B_i z_{f,i}^{k}, s_i^{k} - h_i \right\|^2 - \left\| A_i z_{i}^{k} + B_i z_{f,i}^{k}, s_i^{k} - h_i \right\|^2$$

$$= \frac{\rho_k^i}{2} \left\| s_i^{k} - s_i^{k-1} \right\|^2 - \rho_k^i \left\| A_i z_{i}^{k} + B_i z_{f,i}^{k}, s_i^{k} - h_i \right\|^2.$$ \hfill (41)

To prove the inequality in (40), we define a function $f(s_i) = \langle \lambda_i^k - s_i, s_i \rangle + \frac{\rho_k^i}{2} \left\| s_i \right\|^2$. The gradient of $f(s_i)$ is $\nabla f(s_i) = \lambda_i^k - \beta_i^k s_i - y_i^{k}$ due to (20e). It follows that $f(s_i^{k-1}) - f(s_i^{k}) = \langle y_i^{k}, s_i^{k} - s_i^{k-1} \rangle \geq 0$, as $f(s_i)$ is convex. Therefore, the inequality in (40) holds.

Under the condition $\rho_k^i \geq \sqrt{2} \beta_i^k$, the descent of $\nabla \rho_k^i$ with updating $(z_{i}^{k}, z_{f,i}^{k}, s_i^{k}, y_i^{k})$ can be proved, which means the augmented Lagrangian function is sufficient descent.

Since the function $f(s_i)$, a part of the augmented Lagrangian function, is Lipschitz differentiable with $\beta_i$, and $\lambda_i$ is bounded by $\left[ \lambda_i^k, \infty \right)$, it is obvious that $\nabla f(s_i^{k}) = \lambda_i^k - \beta_i^k s_i = -y_i^{k}$ due to (20c). Thus, it follows that $f(s_i^{k-1}) - f(s_i^{k}) = \langle y_i^{k}, s_i^{k} - s_i^{k-1} \rangle \geq 0$, as $f(s_i)$ is convex. Therefore, the inequality in (40) holds.

5. Update $\lambda_i$ and $\beta_i$. Then, in the outer iterations, the dual variable $\lambda_i^k$ is updated. In order to drive the slack variable $s_i$ to
Suppose that the point $\epsilon\hat{r}$ and condition is proposed, i.e., $\|s_i^k\| > \omega|s_i^{k-1}|$, $\omega \in (0, 1)$.

$$
\lambda_i^{k+1} = \text{Proj}_{[\omega, 1]}(\lambda_i^k + \beta_i^k s_i^k)
$$

$$
\beta^{k+1} = \begin{cases}
\gamma \beta_i^k, & \|s_i^k\| > \omega|s_i^{k-1}| \\
\beta_i^k, & \|s_i^k\| \leq \omega|s_i^{k-1}|,
\end{cases}
$$

where the projection operator $\text{Proj}_{[\omega, 1]}(\lambda^k + \beta^k s_i^k)$ means the projection of $\lambda_i^k + \beta_i^k s_i^k$ onto a closed convex set $[\omega, 1]$.

**Lemma 4.** Given a hypercube set $[a, b]$ with $a, b \in \mathbb{R}^n$ and $a < b$, for any $z \in \mathbb{R}^n$, the projection of $z$ onto the hypercube $[a, b]$ is given by

$$
\text{Proj}_{[a, b]}(z) = \begin{cases}
z_{i(i)}, & a_i \leq z_{i(i)} \leq b_i \\
a_i, & z_{i(i)} < a_i \\
b_i, & z_{i(i)} > b_i,
\end{cases}
$$

where the subscript $\cdot_{(i)}$ means the $i$th component in vector $z$, $a$ and $b$.

**Theorem 2.** Suppose that the point $(z_i^k, z_{f,i}^k, s_i^k, y_i^k)$ optimizes the problem condition (20). For any $\epsilon_1, \epsilon_2, \epsilon_3 > 0$, after finite outer iterations $k$, appropriate stationary point $(z_i^k, z_{f,i}^k, s_i^k, y_i^k)$ can be found such that the optimality condition of the original problem (16) before introducing the slack variable $s_i$ is also satisfied (28), i.e.,

$$
\begin{align*}
0 &\in \nabla J_i(z_i^k) + N_{Z_i(z_i^k)} + A_i^T y_i^k & \quad (44a) \\
0 &\in B_i^T y_i^k + N_{R_i^2(z_{f,i}^k)}(z_{f,i}^k) & \quad (44b) \\
A_i z_i^k + B_i z_{f,i}^k - h_i & = 0. & \quad (44c)
\end{align*}
$$

**Proof.** Assume that $(z_i^k, z_{f,i}^k, s_i^k, y_i^k)$ converges to the stationary point $(z_i^*, z_{f,i}^*, s_i^*, y_i^*)$. We have $z_i^* \in Z_i, z_{f,i}^* \in \mathbb{R}^{2(m+n)}_{+}$ and $A_i z_i^* + B_i z_{f,i}^* - h_i = 0$.

For the case where $\beta_i^k$ is bounded, we have $z_i^k \to 0$ and $z_i^* = 0$. For the case where $\beta_i^k$ is not bounded, based on the optimality condition (20), it follows that

$$
\frac{\lambda_i^k}{\beta_i^k} + s_i + \frac{y_i}{\beta_i^k} = 0.
$$

Then, we have $s_i^* = 0$ by taking the limitation of (45) to zero, as $y_i^k \to y_i^*$ and $\lambda_i^k \in [\omega_i, \tilde{x}_i]$. Since we have $s_i^* = 0$ in both of the two cases, the optimality condition with respect to $y_i$, i.e., (45), can be satisfied. Besides, the optimality condition with respect to $z_i$ and $z_{f,i}$, i.e., (44a) and (44b), can also be satisfied by taking $k \to 0$ on (20a) and (20b), respectively. This completes the proof of Theorem 2.

**Remark 3.** Our algorithm explicitly demonstrates how to apply the hierarchical ADMM in the multi-agent system with the presence of nonconvex collision-avoidance constraints. The closed-form solution of every variable is provided. In this approach, a proximal term is introduced to the two-level algorithm presented in [28] and a more compact convergence condition is proposed, i.e., $\rho_i \geq \sqrt{2}\beta_i$. Therefore, the Theorem 1 in our paper proves the convergence of the inner iteration under the condition $\rho_i \geq \sqrt{2}\beta_i$. Theorem 2 is proved under different assumptions, compared with the algorithm proposed in [28]. Thus, we believe our algorithm and theorems provide more details about the hierarchical ADMM algorithms for the multi-agent systems.

The stopping criterion of this ADMM algorithm is given by

$$
\begin{align*}
\left\|\rho_i^k A_i^T \left(B_i z_{f,i}^r + s_i^r - B_i z_{f,i}^{r+1} - s_i^{r+1}\right)\right\| & \leq \epsilon_1^k & \quad (46a) \\
\left\|\rho_i^k B_i^T (s_i^r - s_i^{r+1})\right\| & \leq \epsilon_2^k & \quad (46b) \\
\left\|A_i z_i^{r+1} + B_i z_{f,i}^{r} - h_i + s_i^{r+1}\right\| & \leq \epsilon_3^k, & \quad (46c)
\end{align*}
$$

where $\epsilon_c^k \to 0, \forall c \in \{1, 2, 3\}$.

To summarize the above developments, the pseudocode of this algorithm is presented in Algorithm 1. After obtaining each agent’s control input and executing the control input, the agents will communicate with their neighbors to obtain the updated position information of their neighbors. Then, this process will be repeated until finishing the designed task.

**Algorithm 1** Hierarchical ADMM for Multi-Agent Decision Making

- Initialize $z_i^0 \in \mathbb{R}^{2(m+n)}, z_{f,i}^0 \in \mathbb{R}^{2T(m+n)}, s_i^0 \in \mathbb{R}^{2Tm+3n}$, initialize the dual variable $\lambda_i^0 \in [\omega_i, \tilde{x}_i], \omega_i, \tilde{x}_i \in \mathbb{R}^{2Tm+3n}$, and $\tilde{x}_i > \omega_i$; initialize the penalty parameter $\beta_i^0 > 0$, and the constant $\omega \in [0, 1)$ and $\gamma > 1$; initialize the sequence of tolerances $(\epsilon_1^k, \epsilon_2^k, \epsilon_3^k) \in \mathbb{R}^+$ with lim$_{k \to \infty} \epsilon_c^k = 0, \forall c \in \{1, 2, 3\}$. Given the terminated tolerances $\epsilon_c, e \in \{1, 2, 3\}$, if $\epsilon_e^k \leq \epsilon_{term,e}$, $\epsilon_e^k = \epsilon_{term,e}$ for $e \in \{1, 2, 3\}$.

Set the outer iteration $k = 0$.

**while** Stopping criterion is violated **do**

- Given $\lambda_i^k$ and $\beta_i^k$, initialize $z_i^{k,0}, z_{f,i}^{k,0}, s_i^{k,0}$ and $y_i^{k,0}$ such that $\lambda_i^k + \beta_i^k s_i^{k,0} + y_i^{k,0} = 0$; initialize tolerance $(\epsilon_1^k, \epsilon_2^k, \epsilon_3^k) \in \mathbb{R}^+$.

- Set the inner iteration $r = 0$.

**while** Stopping criterion is violated **do**

- Update $z_i^{k,r+1}$ by (20) via nonlinear programming solvers.

- Update $z_{f,i}^{k,r+1}, s_i^{k,r+1}, y_i^{k,r+1}$ by (31), (33), (34), respectively.

- $r = r + 1$.

**end while**

- Set $z_i^{k+1}, z_{f,i}^{k+1}, s_i^{k+1}, y_i^{k+1}$ to be $z_i^{k,r+1}, z_{f,i}^{k,r+1}, s_i^{k,r+1}, y_i^{k,r+1}$, respectively.

- Update $\lambda_i^{k+1}$ and $\beta_i^{k+1}$ by (42).

- $k = k + 1$.

**end while**

**V. IMPROVED HIERARCHICAL ADMM**

In the hierarchical ADMM approach, the minimization of $z_i$ in each iteration is computationally expensive, and it requires nonlinear programming (NLP) solvers. Therefore, the improved minimization of $z_i$ is required to improve the computational efficiency by approximating and solving this inexact minimization problem.
The original $z_i$-minimization problem can be expressed as
\[
\min_{z_i} \quad J_i(z_i) + \frac{\theta_i}{2} \left\| A_i z_i + B_i z_{f,i} + s_i - h_i + \frac{y_i^{k,r}}{\rho_i^r} \right\|^2
\]
\[
\text{s. t.} \quad \phi(z_i; [p_j]) \leq 0 \quad \text{and} \quad \psi(z_i; [p_j]) \leq 0,
\]
where
\[
\phi(z_i; [p_j]) = M(z_i; [p_j]) P(z_i; [p_j]) - d^2_{\text{cmu}}
\]
\[
\psi(z_i; [p_j]) = d^2_{\text{safe}} - M(z_i; [p_j]) P(z_i; [p_j]).
\]
The KKT conditions of this $z_i$-minimization problem are
\[
0 = \nabla J_i(z_i) + \rho_i^k A_i^T \left( A_i z_i + B_i z_{f,i} + s_i - h_i + \frac{y_i^{k,r}}{\rho_i^r} \right) + \sum_{\theta=1}^{\theta_\phi} \mu_{\theta,i} \nabla \phi(z_i; [p_j]) + \sum_{\theta=1}^{\theta_\psi} \kappa_{\theta,i} \nabla \psi(z_i; [p_j])
\]
\[
= 0 \quad \forall \theta = 1, 2, \ldots, \theta_\phi
\]
\[
= 0 \quad \forall \theta = 1, 2, \ldots, \theta_\psi.
\]
Therefore, it remains to minimize $\hat{L}_b$, which is given by
\[
\hat{L}_b = J_i(z_i) + \frac{\theta_i}{2} \left\| A_i z_i + B_i z_{f,i} + s_i - h_i + \frac{y_i^{k,r}}{\rho_i^r} \right\|^2 - \langle b_i^T 1_{Tr_i}, \ln (h(z_i; [p_j]) - d^2_{\text{safe}}) \rangle,
\]
where $h(z_i; [p_j]) = M(z_i; [p_j]) P(z_i; [p_j])$.

**Lemma 5.** The gradient of $h(z_i; [p_j])$ with respect to $z_i$ is a linear mapping with respect to $z_i$.

**Proof.** Define $W = I_{Tr_i} \otimes I_{n_p} \in \mathbb{R}^{Tr_i \times n_p \times Tr_i}$, $\omega = (I_T \otimes 1_{r_i} \otimes I_{n_p}) M_p z_i - 1_T \otimes [p_j] \in \mathbb{R}^{Tr_i \times n_p}$, $\Omega = \text{diag}(\omega) \in \mathbb{R}^{Tr_i \times n_p \times Tr_i, n_p}$. Then,
\[
\nabla h(z_i; [p_j]) = W^T (I_{Tr_i} \otimes 1_{r_i} \otimes I_{n_p}) (\omega^T 1_{Tr_i, n_p}) \omega = W^T \Omega \omega.
\]
Since $\nabla h(z_i; [p_j]) = W^T (I_{Tr_i} \otimes 1_{r_i} \otimes I_{n_p}) M_p d z_i$. (51)

It follows
\[
\frac{d h(z_i; [p_j])}{d z_i} = 2 W^T \Omega (I_T \otimes 1_{r_i} \otimes I_{n_p}) M_p d z_i. (52)
\]
Therefore, Lemma 5 is proved.

Set $\Upsilon_i = -\psi(z_i; [p_j])$ and define $\Xi(z_i \otimes \Upsilon_i = 1_{Tr_i}$. Then, the gradient of the barrier augmented Lagrangian function $\hat{L}_b$ is
\[
\nabla \hat{L}_b = 2 H_i(z_i - z_{i,ref}) + \rho_i^k A_i^T \left( A_i z_i + B_i z_{f,i} + s_i - h_i + \frac{y_i^{k,r}}{\rho_i^r} \right) - 2 b_i^T \text{Tr} (1_{Tr_i} \otimes \Xi \otimes W^T \Omega (I_T \otimes 1_{r_i} \otimes I_{n_p}) M_p) .
\]
A closed-form algebraic solution of problem (47) is challenging to determine. However, since the gradient $\nabla \hat{L}_b$ is available, some well-known gradient-based numerical methods, e.g., Barzilai-Borwein, Polak-Ribiere, and Fletcher-Reeves, can be used to calculate the optimal $z_i$. Therefore, the $z_i$-update step in Algorithm 1 can be replaced by solving an unconstrained optimization problem (47) with the known gradient.

Usually, iterative algorithms for NLP need to be called when solving the $z_i$ minimization problem (47), and always searching for a highly accurate solution in each ADMM iteration will result in an excessive computational cost. It is thus desirable to solve the optimization subproblem (47) in ADMM inexactly when the dual variables are yet far from the optimum, i.e., to allow $z_i^{r,r+1}$ to be chosen such that
\[
d^2_{r+1} \in \partial_z L_{b_i}.
\]
In this paper, we use externally shrinking and summable sequence of the absolute errors, i.e.,
\[ \|d_\xi\| \leq c_1^k, \sum_{k=1}^{\infty} c_k^k \leq \infty. \] (55)
such as a summable absolute error sequence can effectively reduce the total number of subroutine iterations throughout the ADMM algorithm. Such a criterion is a constructive one, rendered to guarantee the decrease of a quadratic distance between the intermediate solutions \( z_i^k, z_j^k, s_i^k, y_i^k \) and the optimum \( z_i^0, z_j^0, s_i^0, y_i^0 \).

The above approximate NLP solution is realizable by NLP solvers where the tolerances of the KKT conditions are allowed to be specified by the user, e.g., the IPOPT solver. The pseudocode of the detailed algorithm is shown in Algorithm 2.

**Algorithm 2 Improved Hierarchical ADMM for Multi-Agent Decision Making**

Initializel \( z_0^k \in \mathbb{R}^{T(m+n)}, z^0 \in \mathbb{R}^{T(m+n)}, s_0^k \in \mathbb{R}^{2Tm+3Tn}; \) initialize the dual variable \( \lambda_i^0 \in \left[ \lambda_i^0, \lambda_i^0 \right] \), where \( \lambda_i^0, \lambda_i^0 \in \mathbb{R}^{2Tm+3Tn} \), and \( \lambda_i^0 > \lambda_i^0 \); initialize the penalty parameter \( \beta^0_0 > 0 \), and the constant \( \omega \in [0, 1) \) and \( \gamma > 1 \); initialize the sequence of tolerance \( \{\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4\} \in \mathbb{R}_+^k \) with \( \lim_{k \to \infty} \epsilon_k = 0, \forall e \in \{1, 2, 3, 4\} \).

Given the terminated tolerances \( \epsilon_{\text{term}, e, \varphi} = \epsilon_{\text{term}, e, \varphi} \forall e \in \{1, 2, 3, 4\}, \) if \( \epsilon_k^e \leq \epsilon_{\text{term}, e, \varphi}, \epsilon_k^e \) is equal to \( \epsilon_{\text{term}, e, \varphi} \) for \( e = \{1, 2, 3, 4\} \).

Set the outer iteration \( k = 0 \).

**while** Stopping criterion is violated or \( b_k^e \geq \epsilon_k^e \) do

Given \( \lambda_i^k \) and \( \beta_i^k \), initialize \( \lambda_0^k \), \( \lambda_0^k \), \( k_0 \), \( k_0 \), \( k_0 \), \( k_0 \), \( k_0 \), \( k_0 \), \( k_0 \), \( y_0 \), \( y_i^k \), \( y_i^k \), \( y_i^k \) such that \( \lambda_i^k + \beta_i^k s_i^k + y_i^k = 0 \); initialize tolerance \( \{\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4\} \); \( \rho_k^e \geq \sqrt{2} \beta_k^e \).

Set the inner iteration \( r = 0 \).

**while** Stopping criterion is violated do

Update \( z_i^k, s_i^k, r+1 \) by minimizing \( (49) \).

Update \( z_i^k, r+1 \), \( s_i^k, r+1 \), \( y_i^k, r+1 \) by \( (31), (33), (34) \), respectively.

\( r = r + 1 \).

**end while**

Set \( z_i^k, r+1, s_i^k, r+1, y_i^k, r+1 \) to be \( z_i^k, r+1, s_i^k, r+1, y_i^k, r+1 \), \( y_i^k, r+1, \) respectively.

Update \( \lambda_i^k+1 \) and \( \beta_i^k+1 \) by \( (42) \).

\( k = k + 1 \).

**end while**

In this method, a nonlinear programming problem without constraints is solved with inequality constraints handled by a fixed barrier constant throughout inner iterations. The barrier constant decreases across outer iterations. In the hierarchical ADMM, the NLP solver needs to solve an inequality-constrained NLP in each inner iteration. When the NLP solver, i.e., interior-point algorithm, is called for NLP with inequalities, the barrier constant will go through an iteration inside the solver, and thus every inner iteration will have more loops of internal iteration. By this barrier Lagrangian construction, we integrate the inner loop of the IPOPT with the inner loop of the hierarchical ADMM algorithm. In addition, in the improved hierarchical ADMM, each inner iteration only solves NLP, and an approximate solution is reached without the need of convergence. The accuracy, namely the tolerances of the errors from the KKT conditions, can be controlled when using the interior-point algorithm. Therefore, these tolerances can be set adaptively throughout the iterations (for instance, the tolerances of the 1st, 5th, and 20th iteration can be set to 0.1, 0.01, and 0.001, respectively). By this adaptive tolerance setting, we integrate the outer loop of the IPOPT with the inner loop of the hierarchical ADMM. Therefore, the improved hierarchical ADMM gives less iteration number and computation time than the hierarchical ADMM.

**VI. ILLUSTRATIVE EXAMPLE**

In order to show the effectiveness of the hierarchical ADMM and improved hierarchical ADMM, a multi-UAV system is presented as an application test platform for demonstration of the proposed decision-making framework. The simulations are performed in Python 3.7 environment with processor Intel Xeon(R) E5-1650 v4 @ 3.60GHz. All nonlinear problems are solved by the open source IPOPT solver.

**A. Dynamic Model**

For each quadcopter system, the system model can be expressed in the following state-space representation:
\[ \dot{x} = \begin{bmatrix} \dot{p} \\ \dot{v} \\ \dot{\zeta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} R(\phi, \theta, \psi)^T v \\ -\omega \times v + gR(\phi, \theta, \psi)e \\ W(\phi, \theta, \psi)\omega \\ J^{-1}(\omega \times J\omega) \end{bmatrix} + \tilde{B}(u_{eq} + u), \] (56)
where \( x \in \mathbb{R}^{12} \) and \( u \in \mathbb{R}^4 \) are state and control input vector for the agents, respectively. In this dynamics model, rotor thrusts of each rotor are chosen as control input vector, i.e., \( u = [F_1 F_2 F_3 F_4]^T \).

For each quadcopter system, the system model can be expressed in the following state-space representation:
\[ \dot{x} = \begin{bmatrix} \dot{p} \\ \dot{v} \\ \dot{\zeta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} R(\phi, \theta, \psi)^T v \\ -\omega \times v + gR(\phi, \theta, \psi)e \\ W(\phi, \theta, \psi)\omega \\ J^{-1}(\omega \times J\omega) \end{bmatrix} + \tilde{B}(u_{eq} + u), \] (56)
where \( x \in \mathbb{R}^{12} \) and \( u \in \mathbb{R}^4 \) are state and control input vector for the agents, respectively. In this dynamics model, rotor thrusts of each rotor are chosen as control input vector, i.e., \( u = [F_1 F_2 F_3 F_4]^T \).

In this method, a nonlinear programming problem without constraints is solved with inequality constraints handled by a fixed barrier constant throughout inner iterations. The barrier constant decreases across outer iterations. In the hierarchical ADMM, the NLP solver needs to solve an inequality-constrained NLP in each inner iteration. When the NLP solver, i.e., interior-point algorithm, is called for NLP with inequalities, the barrier constant will go through an iteration inside the solver, and thus every inner iteration will have more loops of internal iteration. By this barrier Lagrangian construction, we integrate the inner loop of the IPOPT with the inner loop of the hierarchical ADMM algorithm. In addition, in the improved hierarchical ADMM, each inner iteration only solves NLP, and an approximate solution is reached without the need of convergence. The accuracy, namely the tolerances of the errors from the KKT conditions, can be controlled when using the interior-point algorithm. Therefore, these tolerances can be set adaptively throughout the iterations (for instance, the tolerances of the 1st, 5th, and 20th iteration can be set to 0.1, 0.01, and 0.001, respectively). By this adaptive tolerance setting, we integrate the outer loop of the IPOPT with the inner loop of the hierarchical ADMM. Therefore, the improved hierarchical ADMM gives less iteration number and computation time than the hierarchical ADMM.

**Here, \( (56) \)** is nonlinear and time-variant, which can be represented as a pseudo-linear form by using state-dependent coefficient factorization \( (51) \). The state-space expression is
\[ x_{t+1} = (A_1 \Delta t + I_{12}) x_t + \tilde{B} \Delta t (u_{eq} + u_t), \] (57)
where \( \Delta t \) is the sampling time interval. Since \( \tilde{A} \) and \( \tilde{B} \) are dependent on the current state \( x \), this state-space representation is in a pseudo-linear form, and then we can suitably consider the system matrices to be constant during the prediction horizon. More details of the parameter settings in \( (56) \) and \((57)\) are presented in \( (32) \).
B. Optimization and Simulation Results

The dynamic model of each agent in the multi-UAV system is shown in Fig. 4. All the agents are set evenly distributed in a circle with the radius of 2 m and \( p_x = 0 \) m, and each agent needs to reach its corresponding point in this same sphere. Therefore, a large number of potential collision possibilities are set up in the simulation task, and our objective is to avoid these designed potential collisions among these agents and make all agents reach their desired destinations. Here, a distributed MPC with a quadratic objective function is designed to realize the path tracking and collision avoidance tasks. Set the number of agents \( N = 8 \). The safety distance \( d_{\text{safe}} \) is 0.2 m. The dimensions are \( n = 12, m = 4, n_p = 3 \). In addition, the agents are connected with all of the remaining agents. The adjacency matrix of the communication network \( D \) is the difference between the upper triangular matrix of the square one matrix \( I_{(N,N)} \) and the identity matrix \( I_N \), i.e., \( D = U(I_{(N,N)}) - I_N \). In this adjacency matrix \( D \), each entry \( D_{ij} \) represents whether there is the connection between the \( i \)th agent and the \( j \)th agent. If \( D_{ij} = 1 \), there is a communication connection between the two agents; otherwise, there is no connection between the two agents. For example, the \( i \)th agent will get communication with the \( i + 1, i + 2, \cdots, N \) agents. The velocity of the agents is confined into \([-5, 5]\) m/s, and the upper and lower bounds of angles \((\phi, \psi, \theta)\) in three dimensions are \((\pi, \frac{\pi}{2}, \pi)\) and \((-\pi, -\frac{\pi}{2}, -\pi)\). The upper bound of control input variable is \( \pi = (1.96, 1.96, 1.96, 1.96) \) N and its lower bound is \( u = -\pi \). Set the prediction horizon \( T = 25 \) with the sampling time \( \Delta t = 0.05 \) s. The weighting matrices \( R_i = 0_{(m,m)} \) and \( Q_i = \text{blockdiag}(I_{n_p}, 0_{(n-n_p,n-n_p)}) \) for all \( i \in \mathcal{V} \). Each element in \( \Lambda_i \) and \( \Lambda_w \) are set as 0.01 and \(-0.01\), respectively. \( \omega \) and \( \gamma \) are set as 0.9 and 1.1. The initial tolerances \( \epsilon^0 \), \( \epsilon^0 \), \( \epsilon^0 \) are defined as \( 10^{-2}, 10^{-2}, 10^{-1} \), respectively, and \( \epsilon^0 = \epsilon^0 / 3^{k-1}, \forall i \in \{1, 2, 3\} \). The terminated tolerances for outer iterations \( \epsilon_{\text{term},1}, \epsilon_{\text{term},2}, \epsilon_{\text{term},3} \) are \( 10^{-6}, 10^{-6}, 10^{-5} \), respectively. The maximum iteration number is set to \( 10^4 \).

The reference trajectories and the resulted trajectories computed by the improved hierarchical ADMM are shown in Fig. 1 and Fig. 2. The initial positions of the 8 agents are distributed uniformly in a circle, whose radius equals 2 m, and the z-axis is 0. In this simulation task, the 8 agents need to move towards their corresponding points, which are the central symmetric points of their initial position, respectively. For example, the agents with initial position \((0, 2, 0) \) m and \((\sqrt{2}, \sqrt{2}, 0) \) m needs to reach \((0, -2, 0) \) m and \((-\sqrt{2}, -\sqrt{2}, 0) \) m. In the two figures, each agent is represented by a different color. These lines in different colors denote the resulted trajectories of the 8 agents, respectively. From the two figures, it is obvious that all agents can effectively avoid their neighbors and maintain the safety distance. On the other hand, for the hierarchical ADMM, a similar performance of collision avoidance can be attained.

In order to show the bound of the control inputs, we only choose the control input results of the 1st agent \( u = [F_1, F_2, F_3, F_4] \) as an example for the clarity of this figure, as shown in Fig. 3. The 4 dimensions of the control inputs are bounded in the predefined range \([-1.96, 1.96]\) N.

The distances between each pair of the 8 agents are shown in Fig. 4. The solid gray line in this figure represents the safety distance all of the agents need to maintain, and the other colored lines denote the distance of each pair of agents with respect to time. It shows that the distance between agents are no smaller than the safety distance. The comparison of computational efficiency between our proposed hierarchical
ADMM and improved hierarchical ADMM can be illustrated, according to Fig. 5 and Fig. 6. Fig. 5 demonstrates the comparison of iteration numbers between the hierarchical ADMM and improved hierarchical ADMM. It is straightforward to observe that the improved hierarchical ADMM can effectively reduce the iteration numbers of both the inner loop and outer loop, compared with the hierarchical ADMM. The comparison of computation time for all inner loop iterations between hierarchical ADMM and improved hierarchical ADMM is shown in Fig. 6. From this figure, it can be observed that the computation time is significantly reduced by using the improved version.

In order to show the effectiveness of the proposed improved hierarchical ADMM, the penalty dual decomposition method [33] and the centralized MPC method are used as the comparison methods. The centralized MPC method means we solve the centralized MPC problem by using a NLP solver, and the penalty dual decomposition method is also a double-loop iterative algorithm [33]. The inner loop of the penalty dual decomposition method is used to solve a nonconvex nonsmooth augmented Lagrangian problem via block-coordinate-descent-type methods, and the outer loop of this algorithm is to update the dual variables and/or the penalty parameter [33]. Table I illustrates the advantages of our proposed algorithms (hierarchical ADMM and improved hierarchical ADMM) over the penalty dual decomposition method and the centralized MPC method.

In order to demonstrate the performance in terms of computational efficiency, the comparison of computation time per step (including all outer loop iterations and inner loop iterations) under the hierarchical ADMM and the improved hierarchical ADMM, and the computation time per step under the centralized MPC (solved by the IPOPT solver) are shown in Fig. 7. From this figure, the computational efficiency of the ADMM for solving large-scale problems can be clearly observed. Moreover, the improved hierarchical ADMM is capable of further improving the efficiency, as compared with the hierarchical ADMM. With the increasing number of agents, the number of nonlinear constraints increases rapidly, which leads to difficulty in solving the problem. Therefore, the computation time is increasing with the increase in the number of agents $N$, as shown in Fig. 8. However, when the centralized MPC solver is used, the growth rate of the computation time is much higher than the ones when the hierarchical ADMM and improved hierarchical ADMM are used, as shown in Fig. 8 which also demonstrates the efficiency of the two proposed methods. Besides, the efficiency of the improved hierarchical ADMM can also be proved in this figure.

VII. CONCLUSION

In this paper, a large-scale nonconvex optimization problem for multi-agent decision making is investigated and solved by the hierarchical ADMM approach. For the multi-agent system, an appropriate DMPC problem is formulated and presented to handle the constraints and achieve the collective objective. Firstly, an innovation using appropriate slack variables is deployed, yielding a resulting transformed optimization problem, which is used to deal with the nonconvex characteristics of the problem. Then, the hierarchical ADMM is applied to solve the transformed problem in parallel. Next, the improved hierarchical ADMM is proposed to increase the computational
efficiency and decrease the number of inner iterations. Additionally, it is analytically shown that the appropriate desired stationary point exists for the procedures of the hierarchical stages for algorithmic convergence. Finally, comparative simulations are conducted using a multi-UAV system as the test platform, which further illustrates the effectiveness and efficiency of the proposed methodology.

**REFERENCES**

[1] X. He, T. Huang, J. Yu, C. Li, and Y. Zhang, “A continuous-time algorithm for distributed optimization based on multiagent networks,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 12, pp. 2708–2709, 2017.

[2] H. Hong, W. Yu, G. Wen, and X. Yu, “Distributed robust fixed-time consensus for nonlinear and disturbed multiagent systems,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1464–1473, 2016.

[3] M. Ye, G. Wen, S. Xu, and F. L. Lewis, “Global social cost minimization with convex objective functions: An extremum seeking-based approach,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020.

[4] J. Ma, S.-L. Chen, C. S. Teo, A. Tay, A. Al Mamun, and K. K. Tan, “Parameter space optimization towards integrated mechatronic design for uncertain systems with generalized feedback constraints,” *Automatica*, vol. 105, pp. 149–158, 2019.

[5] S. Yang, Q. Liu, and J. Wang, “Distributed optimization based on a multiagent system in the presence of communication delays,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 5, pp. 717–728, 2016.

[6] ———, “A multi-agent system with a proportional-integral-derivative protocol for distributed constrained optimization,” *IEEE Transactions on Automatic Control*, vol. 62, no. 7, pp. 3461–3467, 2016.

[7] D. Yuan, S. Xu, and H. Zhao, “Distributed primal–dual subgradient method for multiagent optimization via consensus algorithms,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 6, pp. 1715–1724, 2011.

[8] E. Camponogara and L. B. De Oliveira, “Distributed optimization for model predictive control of linear-dynamic networks,” *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 39, no. 6, pp. 1331–1338, 2009.

[9] M. Hong, “A distributed, asynchronous, and incremental algorithm for nonconvex optimization: an ADMM approach,” *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 935–945, 2017.

[10] J. Ma, Z. Cheng, X. Zhang, M. Tomizuka, and T. H. Lee, “On symmetric Gauss-Seidel ADMM algorithm for H∞ guaranteed cost control with convex parameterization,” *arXiv preprint arXiv:2001.07086*, 2020.

[11] ———, “Optimal decentralized control for uncertain systems by symmetric Gauss-Seidel semi-proximal ALM,” *IEEE Transactions on Automatic Control*, DOI: 10.1109/TAC.2021.3032768, 2021.

[12] Z. He, E. C. Chi, and G. I. Allen, “ADMM algorithmic regularization paths for sparse statistical machine learning,” in *Splitting Methods in Communication, Imaging, Science, and Engineering*, Springer, 2016, pp. 433–459.

[13] J. Ding, Y. Gong, C. Zhang, M. Pan, and Z. Han, “Optimal differentially private ADMM for distributed machine learning,” *arXiv preprint arXiv:1901.02069*, 2019.

[14] B. He and X. Yuan, “On the O(1/n) convergence rate of the Douglas-Rachford alternating direction method,” *SIAM Journal on Numerical Analysis*, vol. 50, no. 2, pp. 700–709, 2012.

[15] R. D. Monteiro and B. F. Svaiter, “Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers,” *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 475-507, 2013.

[16] E. Wei and A. Ozdaglar, “Distributed alternating direction method of multipliers,” in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 5445–5450.

[17] A. Malekzadeh and A. Ozdaglar, “Convergence rate of distributed ADMM over networks,” *IEEE Transactions on Automatic Control*, vol. 62, no. 10, pp. 5082–5095, 2017.

[18] B. He, M. Tao, and X. Yuan, “Alternating direction method with Gaussian back substitution for separable convex programming,” *SIAM Journal on Optimization*, vol. 22, no. 2, pp. 313–340, 2012.

[19] T. Lin, S. Ma, and S. Zhang, “On the global linear convergence of the ADMM with multiblock variables,” *SIAM Journal on Optimization*, vol. 25, no. 3, pp. 1478–1497, 2015.

[20] G. Li and T. K. Pong, “Global convergence of splitting methods for nonconvex composite optimization,” *SIAM Journal on Optimization*, vol. 25, no. 4, pp. 2434–2460, 2015.

[21] H. Wang and A. Banerjee, “Bregman alternating direction method of multipliers,” in *Advances in Neural Information Processing Systems*, 2014, pp. 2816–2824.

[22] B. Jiang, T. Lin, S. Ma, and S. Zhang, “Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis,” *Computational Optimization and Applications*, vol. 72, no. 1, pp. 115–157, 2019.

[23] Y. Wang, W. Yin, and J. Zeng, “Global convergence of ADMM in nonconvex nonsmooth optimization,” *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.

[24] X. Wang, J. Yan, B. Jin, and W. Li, “Distributed and parallel ADMM for structured nonconvex optimization problem,” *IEEE Transactions on Cybernetics*, 2019.

[25] Z. Cheng, J. Ma, X. Zhang, C. W. de Silva, and T. H. Lee, “ADMM-based parallel optimization for multi-agent collision-free model predictive control,” *arXiv preprint arXiv:2101.09894*, 2021.

[26] J. Ma, Z. Cheng, X. Zhang, M. Tomizuka, and T. H. Lee, “Alternating direction method of multipliers for constrained iterative LQR in autonomous driving,” *arXiv preprint arXiv:2110.00462*, 2020.

[27] B. Houska, J. Frasch, and M. Diehl, “An augmented lagrangian based algorithm for distributed nonconvex optimization,” *SIAM Journal on Optimization*, vol. 26, no. 2, pp. 1101–1127, 2016.

[28] K. Sun and X. A. Sun, “A two-level distributed algorithm for nonconvex constrained optimization,” *arXiv preprint arXiv:1902.07654*, 2019.

[29] W. Tang and P. Daoutidis, “Fast and stable nonconvex constrained distributed optimization: The ELLADA algorithm,” *arXiv preprint arXiv:2004.01977*, 2020.

[30] D. Limón, T. Alamo, F. Salas, and E. F. Camacho, “On the stability of constrained mpc without terminal constraint,” *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 832–836, 2006.

[31] X. Zhang, J. Ma, S. Huang, Z. Cheng, and T. H. Lee, “Integrated planning and control for collision-free trajectory generation in 3D environment with obstacles,” in *Proceedings of International Conference on Control, Automation and Systems*, 2019, pp. 974–979.

[32] X. Zhang, J. Ma, Z. Cheng, S. Huang, S. S. Ge, and T. H. Lee, “Trajectory generation by chance-constrained nonlinear MPC with probabilistic prediction,” *IEEE Transactions on Cybernetics*, DOI: 10.1109/TCYB.2020.3032711, 2020.

[33] Q. Shi and M. Hong, “Penalty dual decomposition method for non-smooth nonconvex optimization—Part I: Algorithms and convergence analysis,” *IEEE Transactions on Signal Processing*, vol. 68, pp. 4108–4122, 2020.