Distributed Model Predictive Control via Separable Optimization in Multi-Agent Networks

Ola Shorinwa and Mac Schwager

***Abstract*— We present a distributed model predictive control method which enables a group of agents to compute their control inputs locally, while communicating with their neighbors over a communication network. While many distributed model predictive control methods require a central station for some coordination or computation of the optimization variables, our method does not require a central station, making our approach applicable to a variety of communication network topologies. With our method, each agent solves for its control inputs without solving for the control inputs of other agents, allowing for efficient optimization by each agent, unlike some other distributed methods. Further, our method attains linear convergence to the optimal control inputs in convex model predictive control problems, improving upon the sub-linear convergence rates provided by some other distributed methods such as dual decomposition methods. Moreover, our algorithm provides a closed-loop controller for convex model predictive control problems with affine constraints. We demonstrate our method in both convex and non-convex model predictive control problems in wireless transceiver alignment and satellite deployment, where we show robustness of our method to time delays.**

***Index Terms*— distributed model predictive control, optimal control, distributed optimization, multi-agent networks, separable optimization.**

I. Introduction

The practical effectiveness of model predictive control (MPC) as a means for optimizing online control actions has spurred its application in a variety of problems such as power management and chemical process control [1]–[5], climate regulation in buildings [6]–[13], stabilization of vehicle platoons [14]–[18], network scheduling and bandwidth allocation [19]–[21], and trajectory optimization in robotics [22]–[26]. In all these problems, the control inputs applied by each agent influence other agents, coupling the performance of all agents through the objective and constraint functions or dynamics models in the problem. As such, achieving the desired overall performance requires coordination among all agents. Centralized methods allow for computation of the control inputs of each agent while considering the existing coupling between the agents; however, these methods require all agents to communicate with a central station, which might not be possible in many problems. Moreover, computation of all the control inputs centrally can be particularly difficult in problems with a large number of agents. Distributed model predictive control methods attempt to resolve these issues by enabling each agent to compute its control inputs locally while communicating with its neighbors over a communication network.

Nonetheless, many distributed model predictive control methods still require a central station for coordination of the agents or computation of some auxiliary variables [27]–[31], imposing restrictive constraints on the topology of the communication network between agents. In other distributed model predictive control methods, each agent solves the global optimization problem for its control inputs [32], [33], requiring each agent to know the objective, dynamics, and constraints functions of all agents, information unavailable to each agent in many situations. Eliminating these requirements, we present a distributed model predictive control method where each agent computes its control inputs locally, in parallel with other agents, allowing for efficient optimization especially in problems that require high frequency control updates.

We decompose the model predictive control problem to allow for distributed computation of the control inputs locally by each agent, following a variant of the alternating direction method of multipliers (ADMM) in [34]. In our algorithm, each agent collaborates with those agents which influence its performance, as defined by its objective and constraint functions, communicating with these agents over a communication network to compute its local control inputs. Each agent solves a smaller model predictive control problem locally, involving only its objective functions, dynamics, and constraint functions. We prove linear convergence of the control inputs computed by each agent to the optimal control inputs for convex model predictive control problems, with strongly convex objective functions. As a result, our method provides faster convergence rates which improve upon the sub-linear convergence rates provided by other distributed model predictive control methods.

Many distributed model predictive control methods only consider affine dynamics models for each agent along with convex state and inputs constraints, with the exception of a few [35], [36]. In contrast, our method applies to non-convex model predictive control problems, with non-convex objective and constraint functions. As such, we demonstrate our algorithm in model predictive control of satellites, considering

This work was supported in part by DARPA YFA award D18AP00064 and NSF NRI awards 1830402 and 1925030. *(Corresponding author: Ola Shorinwa).*

O. Shorinwa is with the Department of Mechanical Engineering, Stanford University, Stanford, CA, USA, (email: shorinwa@stanford.edu).

M. Schwager is with the Department of Aeronautics and Astronautics Engineering, Stanford University, Stanford, CA, USA, (email: schwager@stanford.edu).

the non-convex dynamics models and problem constraints, representative of other non-convex model predictive control problems. In addition, we apply our algorithm to a wireless transceiver alignment problem using convex model predictive control, where we demonstrate robustness of our method to time delays. Empirically, our method converges about two times faster in the convex problem and about nine times faster in the non-convex problem compared to other distributed model predictive control methods.

Contributions

Our contributions are as follows:

- We present an algorithm for separable optimization of distributed model predictive control problems (SOD-MPC), where each agent solves a smaller optimization problem locally to compute its control inputs without relying on a central station for any coordination or computation, in contrast with some other distributed methods.
- With our algorithm, each agent does not compute the control inputs of other agents irrelevant to its performance and does not share its objective functions, dynamics models, and constraint functions with other agents, preserving the privacy of each agent.
- In addition, our algorithm produces a closed-loop controller for convex model predictive control problems with affine constraints, providing robustness to modeling errors and time delays arising from computation and communication between agents.

II. Related Works

Lyapunov-based distributed control methods enable agents to solve model predictive control problems locally in a sequential order, with each agent receiving the control inputs of all preceding agents [27], [36]–[39]. Upon computation of its control inputs, each agent communicates its control inputs as well as the control inputs of all preceding agents to the following agent in the sequence. By not allowing for parallel computation of the control inputs by each agent, these methods can be ineffective in problems with a large number of agents, as each agent waits for all preceding agents to compute their control inputs before computing its control inputs. Moreover, the communication complexity of these methods grows linearly with the number of agents, further limiting the scalability of these approaches. These methods assume the existence of a Lyapunov-based controller which might not be readily available in many situations.

In other distributed approaches [32], [33], [35], [40], each agent optimizes over the global optimization problem to compute its control inputs in parallel with other agents, while constraining the control inputs of other agents to the values received from these agents at the beginning of each iteration. Consequently, these methods require each agent to communicate with all other agents in the problem, imposing a stringent requirement on the topology of the communication network. Further, these methods require each agent to have knowledge of the dynamics models, constraint functions, and objective functions of all agents, information not necessarily available to all agents in many problems.

Other approaches employ a distributed hierarchical control scheme where each agent shares its reference trajectory with all its neighbors and solves a local model predictive control problem to compute its control inputs [28], [41]. These approaches combine a stabilizing state-feedback controller, computed centrally, with the nominal control inputs, which can be computed using tube-based methods [42], during online execution of the distributed model predictive control scheme. Centralized computation of a state-feedback controller along with feasible state and control sets can prove challenging, making these methods unsuitable in many problems. In addition, these approaches assume that the actual trajectories of all agents remain within a bounded region of the reference trajectory, a limiting assumption that is violated when agents deviate beyond the bounds of the reference trajectory to satisfy feasibility constraints in their local problems.

Dual decomposition methods enable each agent to compute its control inputs without knowledge of the local objective functions and constraints of other agents by solving the dual problem [43]–[46], obtained from the Lagrangian of the original optimization problem, (refer to [47] for a survey of dual decomposition methods). At each iteration of these methods, each agent computes a dual solution from which the agent can compute its control inputs. However, feasibility of the resulting control inputs only occur after convergence of the dual variables to the optimal dual solution. Consequently, the iterations within dual decomposition methods cannot be terminated at any time before convergence to maintain feasibility, even in cases where the time utilized for computation exceeds the agent's control intervals. Moreover, dual decomposition methods exhibit sub-linear convergence at $O(1/k)$, and thus require a significant number of iterations to converge in many problems. To improve its convergence rate, some dual approaches employ accelerated gradient methods in updating the dual variables [48], while others utilize a primal-dual active-set method [49]. In some other methods, each agent computes its control inputs from the primal problem using a distributed barrier method [50], [51] which ensures feasibility of the computed control inputs.

The alternating direction method of multipliers (ADMM) improves upon dual decomposition methods, extending its convergence properties to a broader class of problems, while retaining its distributed properties. Some distributed model predictive control methods apply ADMM to the dual problem, where each agent keeps a local copy of the dual variable with a consensus constraint on these variables [52], [53]. These methods suffer the same challenges with dual decomposition methods with respect to feasibility of the resulting control inputs and a sub-linear convergence rate of $O(1/k)$. Other methods apply ADMM to the primal problem, overcoming these challenges, but require a central station for updating the dual variables or for termination of the iterations [29]–[31], [54]–[56], making them unsuitable for many problems. To eschew central computation, some primal methods require each agent to compute the control inputs and states of all agents, resulting in local optimization problems with a significantly greater number of optimization variables which require notable

computational effort to solve. In addition, all agents share their solutions with their neighbors, further degrading the efficiency of these methods.

Our distributed model predictive control method overcomes these challenges, enabling each agent to compute its control inputs efficiently, without having to compute the control inputs of other agents unnecessarily. Further, our algorithm applies to problems with a variety of communication network topologies and scales to problems with a large number of agents.

The paper is organized as follows: In Section IV, we present the model predictive control problem along with the dynamics models, objective functions, and problem constraints between agents. In Section V, we derive SOD-MPC, a distributed method for model predictive control problems, and prove linear convergence of our method to the optimal control inputs for strongly convex model predictive control problems. We derive a distributed closed-loop controller in Section VI and provide an efficient procedure for computing the optimal control inputs using the Riccati equations. We demonstrate superior convergence rates of our method compared to dual decomposition and ADMM-MPC methods in Section VII and examine the robustness of the closed-loop controller provided by our approach compared to other distributed approaches and an open-loop centralized controller. We conclude in Section VIII.

III. Preliminaries and Notation

We provide the definition of the following mathematical concepts that will prove useful in the analysis of our algorithm.

Definition 1 (Convex Set)**.** *A set $\mathcal{C}$ is convex if for all $x, y \in \mathcal{C}$ and all $\zeta \in [0,1]$*

$$\zeta x + (1-\zeta)y \in \mathcal{C}. \tag{1}$$

Definition 2 (Convex Function)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if its domain $\mathrm{dom}(f) \subseteq \mathbb{R}^n$ is convex, and for all $x, y \in \mathrm{dom}(f)$ and all $\zeta \in [0,1]$*

$$f(\zeta x + (1-\zeta)y) \leq \zeta f(x) + (1-\zeta) f(y). \tag{2}$$

Definition 3 (Strong Convexity)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}$ is m_f-strongly convex if and only if there exists $m_f > 0$ such that*

$$f(y) \geq f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + m_f \|y-x\|^2, \tag{3}$$

for all $x, y \in \mathrm{dom}(f)$ where $\nabla f(x)$ denotes the gradient of f evaluated at x.

Definition 4 (Lipschitz Continuity)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is Lipschitz continuous at $x \in \mathrm{dom}(f)$ if there exists a constant $L_f > 0$ such that for all $y \in \mathrm{dom}(f)$ sufficiently close to x*

$$\|f(y) - f(x)\|_p \leq L_f \|y-x\|_p, \tag{4}$$

where $\|\cdot\|_p$ denotes the p-norm. When the inequality in (4) holds for all $x, y \in \mathrm{dom}(f)$, f is a Lipschitz function. In this work, we consider the ℓ_2-norm.

Definition 5 (Q-linear Convergence)**.** *A sequence $\{z^k\}$ converges to a stationary point $z^\star$ Q-linearly if there exists $\delta \in (0,1)$ such that*

$$\frac{\|z^{k+1} - z^\star\|}{\|z^k - z^\star\|} \leq \delta \tag{5}$$

for all k sufficiently large.

Definition 6 (R-linear Convergence)**.** *A sequence $\{z^k\}$ converges to a stationary point $z^\star$ R-linearly if there exists another sequence $\{b^k\}$ such that*

$$\|z^{k+1} - z^\star\| \leq b^k \ , \tag{6}$$

$\{b^k\}$ converges Q-linearly to zero, and $b^k \geq 0$ for all k.

We represent the maximum singular value of a matrix A as $\sigma_{\max}(A)$ and its minimum non-zero singular value as $\sigma_{\min}(A)$.

IV. Problem Formulation

We consider a general model predictive control problem among N agents where the performance of each agent influences the performance of other agents through coupling constraint functions, denoted by $\phi_{ij}(\cdot)$ for a coupling constraint between agents i and j. We represent the coupling between agents graphically, as discussed in the following subsection.

Coupling Graph

We represent the coupling between agents using a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a set of vertices $\mathcal{V}$ denoting the agents and a set of edges $\mathcal{E}$ denoting the coupling between agents; hence, if agent i influences the control inputs of agent j, then edge (i,j) exists in $\mathcal{E}$. We assume that an agent can communicate with another agent over a communication network in a timely fashion if its states and control inputs influence the control inputs of the other agent, to ensure feasibility of the control inputs computed by each agent. We describe the neighbor set of agent i as $\mathcal{N}_i$, consisting of all agents which can communicate with agent i.

Figure 1 depicts the graphical representation of the coupling constraints between agents.

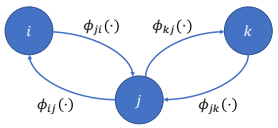

Fig. 1. Graphical representation of the coupling constraints ϕ between agents in the model predictive control problem. The vertices represent the agents while edges between agents signify a coupling constraint between these agents.

Remark 1 (Dynamic Graphs)**.** *Our method allows for dynamic coupling graphs, which can change from time step to time step of the MPC problem; however, we assume the coupling graph remains static within a single time step during which the MPC optimization problem is solved. Empirically, we see solve times on the order of tens of milliseconds, which is typically faster than the timescale of the dynamics, giving a justification for this assumption.*

Coupling between agents in the model predictive control problem could arise through the dynamics model of each agent. We define $\mathcal{N}_i^d \subseteq \mathcal{N}_i$ as the set of agents which influence the dynamics of agent i. We denote the state of agent i at time t as $\hat{x}_{i,t} \in \mathbb{R}^{\hat{n}_i}$, with its control inputs denoted by $\hat{u}_{i,t} \in \mathbb{R}^{\hat{m}_i}$. We represent the dynamics model of agent i by

$$\hat{x}_{i,t+1} = \mathcal{M}_{i,t}(\hat{x}_{i,t}, X_{i,t}, \hat{u}_{i,t}, U_{i,t}) \tag{7}$$

where $X_{i,t} = [\hat{x}_{r,t}^\mathsf{T},\ \forall r \in \mathcal{N}_i^d]^\mathsf{T}$ denotes the concatenation of the states of all agents which influence the dynamics of agent i, with $U_{i,t}$ denoting the concatenation of the control inputs of these agents.

For distributed optimization, agent i keeps a local copy of the variables $X_{i,t}$ and $U_{i,t}$, $\forall t$, with an associated set of equality constraints between corresponding variables computed by agent i and each agent in $\mathcal{N}_i^d$. We denote agent i's local copies of agent j's state and control inputs as $\hat{x}_{j,t}^i$ and $\hat{u}_{j,t}^i$, respectively, and its local copies of the concatenation of the states and control inputs of all agents in $\mathcal{N}_i^d$ as $X_{i,t}^c$ and $U_{i,t}^c$, respectively. With this procedure, the dynamics model of agent i depends on its states and control inputs and its local variables $X_{i,t}^c$ and $U_{i,t}^c$. Further, the dynamics model of agent i induces the equality constraints given by

$$\hat{x}_{j,t}^i = \hat{x}_{j,t},\ \hat{u}_{j,t}^i = \hat{u}_{j,t} \quad \forall t, \quad \forall j \in \mathcal{N}_i^d, \tag{8}$$

which are incorporated into the corresponding coupling constraint functions between agents i and j, given by $\phi_{ij}(\cdot)$, $\forall j \in \mathcal{N}_i^d$.

We represent the concatenation of the state of agent i and $X_{i,t}^c$ at time t as $x_{i,t} = [\hat{x}_{i,t}^\mathsf{T},\ X_{i,t}^{c\mathsf{T}}]^\mathsf{T}$ and the concatenation of its control inputs and $U_{i,t}^c$ as $u_{i,t} = [\hat{u}_{i,t}^\mathsf{T},\ U_{i,t}^{c\mathsf{T}}]^\mathsf{T}$. In addition, we represent the optimization variables of agent i as $x_i = [x_{i,0}^\mathsf{T}, \cdots, x_{i,\mathcal{P}}^\mathsf{T}]^\mathsf{T}$ and $u_i = [u_{i,0}^\mathsf{T}, \cdots, u_{i,\mathcal{P}}^\mathsf{T}]^\mathsf{T}$, concatenating its states and control inputs over a problem horizon $\mathcal{P}$ on each instance of the model predictive control problem. We collect the dynamics constraints across all time indices into a single constraint function $d_i(x_i, u_i) = 0$.

Over the graph $\mathcal{G}$, we describe the model predictive control problem by

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{u}}{\text{minimize}} \ & \sum_{i=1}^{N} f_i(x_i, u_i) \\ \text{subject to } & d_i(x_i, u_i) = 0 \quad i = 1, \cdots, N \\ & g_i(x_i, u_i) = 0 \quad i = 1, \cdots, N \\ & h_i(x_i, u_i) \leq 0 \quad i = 1, \cdots, N \\ & \phi_{ij}(x_i, u_i, x_j, u_j) = 0 \quad \forall (i,j) \in \mathcal{E} \end{aligned} \tag{9}$$

where $x_i \in \mathbb{R}^{n_i}$, $u_i \in \mathbb{R}^{m_i}$, and $f_i : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \to \mathbb{R}$ represents the local objective function of agent i. We assume the objective function f_i is lower-bounded, $\forall i$. We represent other equality constraints in the model predictive control problem by g_i, including constraints on its initial states. Likewise, we include inequality constraints on the optimization variables of agent i in h_i, such as constraints on its feasible states and its control limits. The optimization variables of agent i depend on the optimization variables of agent j through the coupling constraint function $\phi_{ij}(\cdot)$. The coupling constraints between agents can vary over time; however, we do not indicate its dependence on time explicitly. We represent the concatenation of the states of all agents as $\boldsymbol{x} \in \mathbb{R}^{n_x}$ and their control inputs as $\boldsymbol{u} \in \mathbb{R}^{n_u}$. For simplicity of exposition, we consider equality constraints for $\phi_{ij}(\cdot)$; however, our method readily applies to inequality constraints between agents. We note that inequality-constrained problems can be reformulated using auxiliary variables into the same form as (9).

For a model predictive control problem with a problem horizon $\mathcal{P}$, we define the objective function as

$$f_i(x_i, u_i) = \mathcal{W}_{i,\mathcal{P}}(x_{i,\mathcal{P}}) + \sum_{\tau=0}^{\mathcal{P}-1} \mathcal{W}_{i,\tau}(x_{i,\tau}, u_{i,\tau}) \tag{10}$$

where $\mathcal{W}_{i,\tau} : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \to \mathbb{R}$ represents the per-stage objective function at time τ. In many model predictive control problems, $\mathcal{W}_{i,\tau}(\cdot)$ takes a quadratic form given by

$$\begin{aligned} \mathcal{W}_{i,\tau}(x_{i,\tau}, u_{i,\tau}) = & \left(x_{i,\tau} - \tilde{x}_{i,\tau}\right)^\mathsf{T} \mathcal{Q}_{i,\tau} \left(x_{i,\tau} - \tilde{x}_{i,\tau}\right) \\ & + u_{i,\tau}^\mathsf{T} \mathcal{R}_{i,\tau} u_{i,\tau} \end{aligned} \tag{11}$$

where $\tilde{x}_{i,\tau}$ denotes a desired reference state trajectory, $\mathcal{Q}_{i,\tau} \in \mathbb{R}^{n_i} \times \mathbb{R}^{n_i}$ represents a positive semi-definite weight matrix, and $\mathcal{R}_{i,\tau} \in \mathbb{R}^{m_i} \times \mathbb{R}^{m_i}$ represents a positive definite weight matrix. The quadratic objective function in (11) penalizes deviations from the desired state trajectory $\tilde{x}_{i,\tau}$, while minimizing energy consumption. We assume that the terminal cost and terminal set are selected such that stability and recursive feasibility of the model predictive control problem are guaranteed [40], [57], [58]. We note a terminal set can be selected to guarantee stability of the model predictive control problem even in situations where a sub-optimal solution is computed, which is particularly relevant in non-convex problems [59], [60]. Moreover, to preserve separability of the objective and constraint functions in (9), we can synthesize a separable terminal cost function, along with local terminal set constraints, to guarantee stability of the model predictive control problem, as described in [46].

V. Distributed Control

Noting that centralized approaches for solving the model predictive control problem (9) require the computation of the control inputs of all agents at a central station, which can be impractical in many situations, we derive an algorithm for distributed model predictive control, SOD-MPC, which enables each agent to compute its local states and control inputs without computing the states and control inputs of other agents irrelevant to its performance.

We note that some other consensus ADMM methods [44], [61], [62] require the introduction of local copies of all the optimization variables arising in the coupling constraint functions in (9), followed by a subsequent transformation of the coupling constraints to local constraints enforced by each agent. Moreover, the coupling constraints in (9) are replaced by consensus constraints between the original optimization variables and the new local copies. This approach results in greater computation and communication overhead, considering the increase in the number of the optimization variables in the

problem. As a result, we derive a distributed algorithm that avoids this approach.

We apply the SOVA method [34] based on ADMM to obtain distributed update procedures for the local variables of each agent. We express the coupling constraint function between agents i and j as

$$\phi_{ij}(x_i, u_i, x_j, u_j) = \phi_{ij}^i(x_i, u_i) - \phi_{ij}^j(x_j, u_j) \tag{12}$$

where $\phi_{ij}^i : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \to \mathbb{R}^{n_{ij}}$ depends only on the optimization variables of agent i and $\phi_{ij}^j : \mathbb{R}^{n_j} \times \mathbb{R}^{m_j} \to \mathbb{R}^{n_{ij}}$ depends only on the optimization variables of agent j. We emphasize that the decomposition in (12) exists for any arbitrary coupling constraint function ϕ_{ij}, which we demonstrate in the subsequent discussion, where we consider two broad classes of model predictive control problems.

Convex Model Predictive Control Problems

In convex model predictive control problems, the coupling constraint functions must be affine, which shows that the optimization variables of agents i and j do not appear together in the same term within the coupling constraint function. Consequently, we can group all terms depending on the optimization variables of agent i into a new function $\phi_{ij}^i(\cdot)$ and, likewise, group all terms depending on the optimization variables of agent j into $\phi_{ij}^j(\cdot)$, such that we obtain the expression in (12). We can assign the constant term in the coupling constraint function to either $\phi_{ij}^i(\cdot)$ or $\phi_{ij}^j(\cdot)$ or split it between both functions.

Non-Convex Model Predictive Control Problems

The coupling constraints in some non-convex model predictive control problems possess a separable structure naturally. For example, coupling constraints of the form

$$\phi_{ij}(u_i, u_j) = \|u_i\|_2^2 - \|u_j\|_2^2 \tag{13}$$

can be simply expressed in the form given in (12) with

$$\phi_{ij}^i(u_i) = \|u_i\|_2^2 \quad \text{and} \quad \phi_{ij}^j(u_j) = \|u_j\|_2^2. \tag{14}$$

We consider more general coupling constraint functions which may not possess a separable structure, e.g., constraint functions with bilinear terms. Many decomposition strategies exist for expressing these constraint functions in the form given in (12). Here, we discuss the simplest decomposition strategy, which involves introducing new optimization variables representing local copies of the original optimization variables in the original coupling constraint. Subsequently, we transform the coupling constraint to a local constraint, enforced by each agent locally. Lastly, we create a new coupling constraint enforcing equality between the original optimization variables and the local copies of these variables which were introduced to decouple the original coupling constraint. The new coupling constraint exists in the form given in (12).

To provide a concrete illustration, we consider the bilinear constraint

$$x_i^\mathsf{T} x_j = 0. \tag{15}$$

We begin by introducing a local copy of x_i, denoted by $\breve{x}_i$, and a local copy of x_j, denoted by $\breve{x}_j$. Next, we assign $\breve{x}_j$ to agent i and $\breve{x}_i$ to agent j. Agent i enforces the local constraint

$$x_i^\mathsf{T} \breve{x}_j = 0, \tag{16}$$

while agent j enforces the local constraint

$$\breve{x}_i^\mathsf{T} x_j = 0. \tag{17}$$

Lastly, we create a pair of new coupling constraints

$$x_i - \breve{x}_i = 0 \quad \text{and} \quad x_j - \breve{x}_j = 0, \tag{18}$$

which are in the form given in (12). We note that more efficient decomposition strategies exist; however, we do not discuss these strategies here, due to space constraints.

Expressing the coupling constraints in the form in (12) enables us to decouple the contribution of each agent to the value of the coupling constraint, allowing each agent to optimize over this constraint locally.

With the constraint in (12), the model predictive control problem in (9) is given by

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{u}}{\text{minimize}} \; & \sum_{i=1}^N f_i(x_i, u_i) \\ \text{subject to } & d_i(x_i, u_i) = 0 \quad i = 1, \cdots, N \\ & g_i(x_i, u_i) = 0 \quad i = 1, \cdots, N \\ & h_i(x_i, u_i) \leq 0 \quad i = 1, \cdots, N \\ & \phi_{ij}^i(x_i, u_i) = \phi_{ij}^j(x_j, u_j) \quad \forall (i,j) \in \mathcal{E} \end{aligned} \tag{19}$$

where agent i computes its local optimization variables x_i and u_i. We introduce the auxiliary variables $c_{ij} \in \mathbb{R}^{n_{ij}}$ in the coupling constraint $\phi_{ij}(\cdot)$, expressing the model predictive control problem in (19) as

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{u},\boldsymbol{c}}{\text{minimize}} \; & \sum_{i=1}^N f_i(x_i, u_i) \\ \text{subject to } & d_i(x_i, u_i) = 0 \quad i = 1, \cdots, N \\ & g_i(x_i, u_i) = 0 \quad i = 1, \cdots, N \\ & h_i(x_i, u_i) \leq 0 \quad i = 1, \cdots, N \\ & \phi_{ij}^i(x_i, u_i) = c_{ij} \quad \forall\ (i,j) \in \mathcal{E} \\ & \phi_{ij}^j(x_j, u_j) = c_{ij} \quad \forall (i,j) \in \mathcal{E} \end{aligned} \tag{20}$$

where $\boldsymbol{c} = [c_{ij}^\mathsf{T},\ \forall (i,j) \in \mathcal{E}]^\mathsf{T}$.

Next, we derive update procedures for the states and control inputs of each agent from the augmented Lagrangian for the problem in (20)

$$\begin{aligned} \mathcal{L}_a(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{c}, \boldsymbol{v}, \boldsymbol{w}) = & \sum_{i=1}^N f_i(x_i, u_i) \\ & + \sum_{(i,j)\in\mathcal{E}} \big(v_{ij}^\mathsf{T}(\phi_{ij}^i(x_i, u_i) - c_{ij}) \\ & \qquad + w_{ij}^\mathsf{T}(\phi_{ij}^j(x_j, u_j) - c_{ij}) \big) \\ & + \frac{\rho}{2} \sum_{(i,j)\in\mathcal{E}} \Big(\|\phi_{ij}^i(x_i, u_i) - c_{ij}\|_2^2 \\ & \qquad + \|\phi_{ij}^j(x_j, u_j) - c_{ij}\|_2^2 \Big) \end{aligned} \tag{21}$$

with dual variables $v_{ij} \in \mathbb{R}^{n_{ij}}$ and $w_{ij} \in \mathbb{R}^{n_{ij}}$ for the coupling constraints between agents i and j, where $\boldsymbol{v} = [v_{ij}^{\mathsf{T}},\ \forall (i,j) \in \mathcal{E}]^{\mathsf{T}}$ and $\boldsymbol{w} = [w_{ij}^{\mathsf{T}},\ \forall (i,j) \in \mathcal{E}]^{\mathsf{T}}$. We have not dualized the dynamics constraints and the local equality and inequality constraints in (21). Rather, we restrict the domain of $\mathcal{L}_a(\cdot)$ such that

$$\begin{aligned} d_i(x_i, u_i) &= 0 \quad i = 1, \cdots, N \\ g_i(x_i, u_i) &= 0 \quad i = 1, \cdots, N \\ h_i(x_i, u_i) &\leq 0 \quad i = 1, \cdots, N \end{aligned} \tag{22}$$

which ensures that the control inputs generated by our algorithm remain feasible for each agent.

The positive parameter ρ in the augmented Lagrangian determines the penalty on the violation of the coupling constraints between the agents. Each agent updates its primal variables iteratively as the minimizers of the augmented Lagrangian using its dual variables at the previous iteration and subsequently updates its dual variables through dual ascent on the augmented Lagrangian using its computed primal variables. We update the primal variables $\boldsymbol{x}$ and $\boldsymbol{u}$ before updating the auxiliary variable $\boldsymbol{c}$.

For any value of $\rho > 0$, the augmented Lagrangian is quadratic in c_{ij}. As a result, the update procedure for c_{ij} yields a closed-form solution, given by

$$c_{ij}^{k+1} = \frac{v_{ij}^k + w_{ij}^k}{2\rho} + \frac{\phi_{ij}^i(x_i^{k+1}, u_i^{k+1}) + \phi_{ij}^j(x_j^{k+1}, u_j^{k+1})}{2}, \tag{23}$$

at iteration k. The dual ascent procedure for updating the dual variables results in

$$v_{ij}^{k+1} = v_{ij}^k + \rho \left(\phi_{ij}^i(x_i^{k+1}, u_i^{k+1}) - c_{ij}^{k+1}\right) \tag{24}$$

and

$$w_{ij}^{k+1} = w_{ij}^k + \rho \left(\phi_{ij}^j(x_j^{k+1}, u_j^{k+1}) - c_{ij}^{k+1}\right). \tag{25}$$

By simplifying (24) and (25) using (23), we obtain that $v_{ij}^{k+1} = -w_{ij}^{k+1}$, at each iteration k.

Agent i updates x_i^k and u_i^k at iteration k as the minimizer of the problem

$$\begin{aligned} \underset{x_i, u_i}{\text{minimize}}\ & f_i(x_i, u_i) + q_i^{k\mathsf{T}} \phi_i^i(x_i, u_i) \\ & + \rho \sum_{j \in \mathcal{N}_i} \left\| \phi_{ij}^i(x_i, u_i) - \psi_{ij}\left(x_i^k, u_i^k, x_j^k, u_j^k\right) \right\|_2^2 \\ \text{subject to}\ & d_i(x_i, u_i) = 0 \\ & g_i(x_i, u_i) = 0 \\ & h_i(x_i, u_i) \leq 0 \end{aligned} \tag{26}$$

where $\phi_i(\cdot)$ represents the vertical concatenation of the coupling constraint functions between agent i and all its neighbors, given by

$$\begin{aligned} \phi_i(\cdot) &= [(\phi_{ij}^i(\cdot) - \phi_{ij}^j(\cdot))^{\mathsf{T}},\ \forall j \in \mathcal{N}_i]^{\mathsf{T}}, \\ \phi_i^i(\cdot) &= [\phi_{ij}^i(\cdot)^{\mathsf{T}},\ \forall j \in \mathcal{N}_i]^{\mathsf{T}}, \end{aligned} \tag{27}$$

and

$$\psi_{ij}\left(x_i^k, u_i^k, x_j^k, u_j^k\right) = \frac{\phi_{ij}^i\left(x_i^k, u_i^k\right) + \phi_{ij}^j\left(x_j^k, u_j^k\right)}{2}. \tag{28}$$

We have introduced a composite dual variable $q_i \in \mathbb{R}^{n_c}$, with the component of q_i^k corresponding to edge (i,j) given by

$$\left[q_i^k\right]_{ij} = v_{ij}^k + w_{ji}^k, \tag{29}$$

where $\boldsymbol{v}^0 = \boldsymbol{w}^0 = 0$. From (24) and (25), we obtain the corresponding update procedure for q_i at iteration k, given by

$$q_i^{k+1} = q_i^k + \rho\ \phi_i\left(x_i^{k+1}, u_i^{k+1}, x_{\mathcal{N}_i}^{k+1}, u_{\mathcal{N}_i}^{k+1}\right) \tag{30}$$

where $x_{\mathcal{N}_i}$ and $u_{\mathcal{N}_i}$ denote the vertical concatenation of the optimization variables of all the neighbors of agent i. In deriving the update procedure in (26), we have assumed that agents i and j have access to c_{ij}^{k+1}, i.e., agent i has access to the value of $\phi_{ij}^j(x_j^{k+1}, u_j^{k+1})$ and agent j has access to the value of $\phi_{ij}^i(x_i^{k+1}, u_i^{k+1})$, which requires that the communication graph within a single instance of the model predictive control problem remains static. This assumption enables us to reduce the number of optimization variables maintained by each robot, as each agent maintains a smaller composite dual variable, as opposed to maintaining a dual variable for each edge in $\mathcal{G}$. Note that this assumption does not restrict the communication graph from changing across different instances of the model predictive control problem. In addition, we note that the update procedures can be modified to allow for dynamic communication graphs within a single instance of the model predictive control problem; however, the resulting algorithm would require each agent to maintain a dual variable for each edge in $\mathcal{G}$.

Each agent does not need to compute the auxiliary variables as these variables have been included in the update procedures in (26). From the update equations (26) and (30), agent i only communicates with its neighbors to evaluate its coupling constraints $\phi_i(\cdot)$ and $\psi_{ij}(\cdot)$, $\forall j \in \mathcal{N}_i$. Specifically, agent i shares the value of $\phi_{ij}^i(x_i^{k+1}, u_i^{k+1})$ with agent j, its neighbor, $\forall j \in \mathcal{N}_i$, at iteration k. As a result, SOD-MPC does not necessarily require each agent to share its iterates for the local optimization variables x, u, and q with its neighbors. Rather, each agent shares the value of its component of the coupling constraints with its neighbors, ensuring the privacy of its local variables. In addition, each agent only requires its local objective and constraint functions in its update procedures, maintaining the privacy of these local functions and the problem data.

Theorem 1. *For a non-convex model predictive control problem, the iterates (x_i^k, u_i^k) of agent i converge to a locally optimal solution, $\forall i \in \mathcal{V}$, with the assumption that the objective function of* (9) *has Lipschitz continuous gradients.*

Proof. We omit the proof here. Refer to [63] for the proof. □

Algorithm 1 outlines SOD-MPC for distributed model predictive control.

At each iteration, the states and control inputs of agent i, contained in its local optimization variables x_i and u_i satisfy all the constraints in (9), except its coupling constraint, which is satisfied at convergence of our algorithm. Hence, the *DistributedControl* procedure can be terminated early to allow each agent to apply its control inputs, if absolute satisfaction of the coupling constraints is not required. Since the iterates

Algorithm 1: Separable Optimization for Distributed Model Predictive Control (SOD-MPC)

do in parallel $i = 1, \cdots, N$
  $(x_i, u_i) \leftarrow \text{DistributedControl}(t, \bar{x}_{i,t})$
  Apply control input $\breve{u}_{i,t}$.
while *task is in progress*;

Function DistributedControl$(t, \bar{x}_{i,t})$

`Initialization:`
  $k \leftarrow 0$
  $q_i^0 \leftarrow 0$
  $(x_i^0, u_i^0) \leftarrow \underset{x_i, u_i}{\text{argmin}}\ f_i(x_i, u_i)$
do in parallel $i = 1, \cdots, N$
  $\left(x_i^{k+1}, u_i^{k+1}\right) \leftarrow$ Procedure (26)
  `Communication Step:`
    Agent i shares the value of $\phi_{ij}^i(x_i^{k+1}, u_i^{k+1})$ with agent j, its neighbor, and receives the value of $\phi_{ij}^j(x_j^{k+1}, u_j^{k+1})$, $\forall j \in \mathcal{N}_i$.
  $q_i^{k+1} \leftarrow$ Procedure (30)
  $k \leftarrow k + 1$
while *not converged or stopping criterion is not met*;
return (x_i, u_i)

of each agent satisfy its dynamics constraints, control inputs constraints, and terminal set constraints, we note that early termination of our algorithm does not negatively impact the stability guarantees of sub-optimal model predictive control [59]. Moreover, our algorithm preserves the same stability guarantees in non-convex model predictive control problems, where, generally, sub-optimal solutions are computed. With SOD-MPC, agent i computes its control inputs $\breve{u}_{i,t}$ at its current state $\bar{x}_{i,t}$ and repeats this procedure for its control inputs at the next time instant.

A. Convergence Analysis for Convex Model Predictive Control Problems

We examine the convergence rate of our algorithm on convex model predictive control problems. We note that the constraint functions d_i, g_i, and ϕ_{ij} are restricted to affine functions in convex model predictive problems. In addition, the function h_i must be convex. We express the optimization problem in (19) in terms of the control inputs by eliminating the state variables using the dynamics model in $d_i(\cdot)$ for agent i. In addition, we eliminate constraints on the initial state of each agent, specified in g_i. With this formulation, the convex model predictive control problem is given by

$$\begin{aligned} \underset{\boldsymbol{u}}{\text{minimize}} & \sum_{i=1}^{N} \mathbb{F}_i(u_i) \\ \text{subject to } & \mathbb{G}_i(u_i) = 0 \quad i = 1, \cdots, N \\ & \mathbb{H}_i(u_i) \leq 0 \quad i = 1, \cdots, N \\ & \Phi_{ij}^i(u_i) = \Phi_{ij}^j(u_j) \quad \forall (i,j) \in \mathcal{E} \end{aligned} \tag{31}$$

where the constraint functions $\mathbb{G}_i$ and Φ_{ij} are affine and the objective function $\mathbb{F}_i$ and constraint function $\mathbb{H}_i$ are convex functions. We assume that the problem in (31) and the associated update procedures are feasible and, in addition, an optimal primal-dual solution of (31) exists.

For convex model predictive problems, described by (31), we provide a general convergence result in the following theorem.

Theorem 2 (Convergence of $\{\boldsymbol{u}^k\}$)**.** *In convex model predictive control problems, the sequence of control inputs computed by all agents $\{\boldsymbol{u}^k\}$ converges sub-linearly to the optimal control inputs $\boldsymbol{u}^\star$.*

Proof. We refer readers to [64] for the proof. □

SOD-MPC converges for any value of $\rho > 0$. Specific strategies for selecting the value of ρ can be found in [65], [66]. We can make stronger statements on the convergence rate of our algorithm in convex model predictive control problems without the local constraint functions $\mathbb{G}_i$ and $\mathbb{H}_i$ in (31). Particularly, the iterates generated by our algorithm converge linearly to the optimal solution in these problems. To show linear convergence of our algorithm, we assume that the local objective function $\mathbb{F}_i$ of agent i is continuously differentiable with Lipschitz continuous gradients and strongly convex, for all agents. This assumption is not restrictive, considering that, in many model predictive control problems, the objective function takes a quadratic form with a positive definite Hessian. In addition, we define the sequence $\{z^k\}$ as

$$z^k = \begin{bmatrix} q^k \\ c^k \end{bmatrix}, \tag{32}$$

composed of the primal variable c and the dual variable q, where $c = [c_{ij}^\mathsf{T},\ \forall (i,j) \in \mathcal{E}]^\mathsf{T}$ and $q = [q_i^\mathsf{T},\ \forall i \in \mathcal{V}]^\mathsf{T}$.

Lemma 1 (Sufficient Descent)**.** *The error $\|z^k - z^\star\|$ decreases monotonically at each iteration, until convergence of the sequence $\{z^k\}$ to a limit point, with*

$$\begin{aligned} &\left\|z^k - z^\star\right\|_W^2 - \left\|z^{k+1} - z^\star\right\|_W^2 \\ &\qquad \geq m_f \left\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^\star\right\|_2^2 + \left\|z^{k+1} - z^k\right\|_W^2, \end{aligned} \tag{33}$$

where $z^\star$ and $\boldsymbol{u}^\star$ denote the optimal solution of z and the control inputs, respectively, and $W > 0$.

Proof. We provide the proof in the Appendix. □

Lemma 2 (Q-linear Convergence of $\{z^k\}$)**.** *The sequence $\{z^k\}$ converges Q-linearly to $z^\star$ with*

$$\frac{\|z^{k+1} - z^\star\|_W^2}{\|z^k - z^\star\|_W^2} \leq \frac{1}{\ell + 1} \tag{34}$$

where

$$\ell = \min\left\{ \frac{4 m_f \rho (\mu - 1) \sigma_{\min}^2(\mathcal{J})}{\rho^2 (\mu - 1) \sigma_{\max}^2(\bar{\mathcal{J}})\, \sigma_{\min}^2(\mathcal{J}) + 4\mu L_f^2}, \frac{\sigma_{\min}^2(\mathcal{J})}{\mu \sigma_{\min}^2(\bar{\mathcal{J}})} \right\} \tag{35}$$

with $\ell > 0$ and $\mu > 1$.

Proof. We provide the proof in the Appendix. □

Theorem 3 (R-linear Convergence of $\{\boldsymbol{u}^k\}$)**.** *The sequence of control inputs computed by all agents $\{\boldsymbol{u}^k\}$ converges R-linearly to the optimal control inputs $\boldsymbol{u}^\star$ with*

$$\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^\star\|_W^2 \leq \frac{1}{m_f}\|z^k - z^\star\|_W^2 \tag{36}$$

where $m_f > 0$, for convex model predictive control problems with strongly convex objective functions and Lipschitz continuous gradients.

Proof. From Lemma 1 and Q-linear convergence of $\{z^k\}$ (Lemma 2),

$$\left\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^\star\right\|_2^2 \leq \frac{1}{m_f}\left\|z^k - z^\star\right\|_W^2 \ , \tag{37}$$

showing R-linear convergence of $\{\boldsymbol{u}^k\}$ to the optimal control inputs $\boldsymbol{u}^\star$. □

VI. Closed-Loop Control Law

SOD-MPC provides a time-varying closed-loop control law for convex model predictive control problems with affine constraints, from the update procedure in (26). We consider model predictive problems where the dynamics model of agent i is given by

$$x_{i,\tau+1} = A_{i,\tau}x_{i,\tau} + B_{i,\tau}u_{i,\tau} \tag{38}$$

with $x_{i,\tau} \in \mathbb{R}^{n_i}$ denoting its state, $u_{i,\tau} \in \mathbb{R}^{m_i}$ denoting its control inputs, and $A_{i,\tau} \in \mathbb{R}^{n_i \times n_i}$ and $B_{i,\tau} \in \mathbb{R}^{n_i \times m_i}$ describing its dynamics, at time τ. In these problems, the agents compute their states and control inputs by solving the optimization problem

$$\begin{aligned}
&\underset{\boldsymbol{x},\boldsymbol{u}}{\text{minimize}} \sum_{i=1}^{N} f_i(x_i, u_i) \\
&\text{subject to } x_{i,0} = \bar{x}_{i,t} \quad i = 1, \cdots, N \\
&\qquad x_{i,\tau+1} = A_{i,\tau}x_{i,\tau} + B_{i,\tau}u_{i,\tau} \quad \forall \tau,\ i = 1, \cdots, N \\
&\qquad \phi_{ij}(x_i, u_i, x_j, u_j) = 0 \quad \forall (i,j) \in \mathcal{E}
\end{aligned} \tag{39}$$

where $\bar{x}_{i,t}$ indicates the state of agent i at time t and f_i represents the objective function of agent i given by

$$f_i(x_i, u_i) = x_{i,\mathcal{P}}^\mathsf{T} H_i x_{i,\mathcal{P}} + \sum_{\tau=0}^{\mathcal{P}-1}\left(x_{i,\tau}^\mathsf{T} Q_{i,\tau} x_{i,\tau} + u_{i,\tau}^\mathsf{T} R_{i,\tau} u_{i,\tau}\right) \tag{40}$$

with a terminal weight $H_i \in \mathbb{R}^{n_i \times n_i}$, weights on agent i's states $Q_i \in \mathbb{R}^{n_i \times n_i}$, and weights on its control inputs $R_i \in \mathbb{R}^{m_i \times m_i}$. For the existence of a stabilizing solution to the model predictive control problem, we require positive definite weights Q_i and positive semi-definite weights for R_i.

Using Algorithm 1, agent i computes its states and control inputs at time t from the optimization problem

$$\begin{aligned}
\underset{x_i,u_i}{\text{minimize}}\ & x_{i,\mathcal{P}}^\mathsf{T} H_i x_{i,\mathcal{P}} + \sum_{\tau=0}^{\mathcal{P}-1}\left(x_{i,\tau}^\mathsf{T} Q_{i,\tau} x_{i,\tau} + u_{i,\tau}^\mathsf{T} R_{i,\tau} u_{i,\tau}\right) \\
&+ q_i^{k\mathsf{T}}\phi_i^i(x_i, u_i) \\
&+ \rho \sum_{j \in \mathcal{N}_i}\left\|\phi_{ij}^i(x_i, u_i) - \psi_{ij}\left(x_i^k, u_i^k, x_j^k, u_j^k\right)\right\|_2^2 \\
\text{subject to } & x_{i,0} = \bar{x}_{i,t} \\
& x_{i,\tau+1} = A_{i,\tau}x_{i,\tau} + B_{i,\tau}u_{i,\tau} \quad \forall \tau
\end{aligned} \tag{41}$$

with $\psi_{ij}(\cdot)$ given by (28). A direct approach to computing agent i's control inputs involves solving the quadratic program (41) numerically. Numerical quadratic optimization techniques involve factorizing the Hessian of the Lagrangian of (41) which can require notable computational effort, depending on the length of the problem horizon and the dimensions of the state and control inputs of each agent. Consequently, this approach quickly creates computation overhead, especially in problems with high control bandwidths.

In contrast, we derive a time-varying closed-loop control law for computing the optimal control inputs of (41), which involves factorizing a matrix of a much smaller dimension, significantly reducing the computation overhead associated with solving (41). For convex model predictive control problems, we eliminate the state variables of agent i in the coupling constraints $\phi_i^i(\cdot)$ and express the resulting coupling constraints at time t as

$$\phi_i^i(u_i, t) = \nu_{i,t}^i u_i + \eta_{i,t}^i \tag{42}$$

where $\nu_{i,t}^i$ represents the Jacobian of $\phi_i^i(\cdot)$ with respect to u_i given by

$$\nu_{i,t}^i = \nabla_{u_i}\phi_i^i(u_i, t) \tag{43}$$

and $\eta_{i,t}^i$ represents constant terms in $\phi_i^i(\cdot)$, independent of agent i's control inputs.

From dynamic programming, agent i solves for its control input $u_{i,\mathcal{P}-1}$ from the optimization problem

$$\begin{aligned}
\underset{x_{i,\mathcal{P}},u_{i,\mathcal{P}-1}}{\text{minimize}}\ & x_{i,\mathcal{P}}^\mathsf{T} H_i x_{i,\mathcal{P}} + q_{i,\mathcal{P}}^{k\mathsf{T}}\nu_{i,\mathcal{P}}^i u_{i,\mathcal{P}-1} \\
&+ \rho \sum_{j \in \mathcal{N}_i}\left\|\nu_{ij,\mathcal{P}}^i u_{i,\mathcal{P}-1} - \gamma_{ij,\mathcal{P}}\left(u_{i,\mathcal{P}-1}^k, u_{j,\mathcal{P}-1}^k\right)\right\|_2^2 \\
&+ x_{i,\mathcal{P}-1}^\mathsf{T} Q_{i,\mathcal{P}-1} x_{i,\mathcal{P}-1} \\
&+ u_{i,\mathcal{P}-1}^T R_{i,\mathcal{P}-1} u_{i,\mathcal{P}-1} + q_{i,\mathcal{P}-1}^{k\mathsf{T}}\nu_{i,\mathcal{P}-1}^i u_{i,\mathcal{P}-1} \\
&+ \rho \sum_{j \in \mathcal{N}_i}\left\|\nu_{ij,\mathcal{P}-1}^i u_{i,\mathcal{P}-1} - \gamma_{ij,\mathcal{P}-1}\left(u_{i,\mathcal{P}-1}^k, u_{j,\mathcal{P}-1}^k\right)\right\|_2^2 \\
\text{subject to } & x_{i,\mathcal{P}} = A_{i,\mathcal{P}-1}x_{i,\mathcal{P}-1} + B_{i,\mathcal{P}-1}u_{i,\mathcal{P}-1}
\end{aligned} \tag{44}$$

given $x_{i,\mathcal{P}-1}$, the state of agent i at time $\mathcal{P}-1$, with

$$\gamma_{ij,\tau}(u_i, u_j) = \frac{\phi_{ij}^i(u_i, \tau) + \phi_{ij}^j(u_j, \tau)}{2} - \eta_{ij,\tau}^i \tag{45}$$

which includes the coupling constraints between agents i and

j. The optimization problem simplifies to

$$\begin{aligned}\underset{x_{i,\mathcal{P}},u_{i,\mathcal{P}-1}}{\text{minimize}}\ & x_{i,\mathcal{P}}^{\mathsf{T}}H_i x_{i,\mathcal{P}} + x_{i,\mathcal{P}-1}^{\mathsf{T}}Q_{i,\mathcal{P}-1}x_{i,\mathcal{P}-1} \\ & + u_{i,\mathcal{P}-1}^{\mathsf{T}}\mathcal{R}_{i,\mathcal{P}-1}u_{i,\mathcal{P}-1} + z_{i,\mathcal{P}-1}^{k\mathsf{T}}u_{i,\mathcal{P}-1} \\ \text{subject to}\ & x_{i,\mathcal{P}} = A_{i,\mathcal{P}-1}x_{i,\mathcal{P}-1} + B_{i,\mathcal{P}-1}u_{i,\mathcal{P}-1}\end{aligned} \tag{46}$$

where

$$\mathcal{R}_{i,\mathcal{P}-1} = R_{i,\mathcal{P}-1} + \rho\sum_{j\in\mathcal{N}_i}\left(\nu_{ij,\mathcal{P}}^{i\mathsf{T}}\nu_{ij,\mathcal{P}}^{i} + \nu_{ij,\mathcal{P}-1}^{i\mathsf{T}}\nu_{ij,\mathcal{P}-1}^{i}\right) \tag{47}$$

and

$$z_{i,\mathcal{P}-1}^{k} = \sum_{\tau=\mathcal{P}-1}^{\mathcal{P}}\left(\nu_{i,\tau}^{i\mathsf{T}}q_{i,\tau}^{k} - 2\rho\sum_{j\in\mathcal{N}_i}\nu_{ij,\tau}^{i\mathsf{T}}\gamma_{ij,\tau}\left(u_{i,\mathcal{P}-1}^{k}, u_{j,\mathcal{P}-1}^{k}\right)\right) \tag{48}$$

for the control inputs $u_{i,\mathcal{P}-1}$ at iteration k. The optimization problem for the control inputs of agent i at time τ takes the same form as the problem in (46), given by

$$\begin{aligned}\underset{x_{i,\tau},u_{i,\tau-1}}{\text{minimize}}\ & x_{i,\tau}^{\mathsf{T}}\mathcal{H}_{i,\tau}x_{i,\tau} + x_{i,\tau-1}^{\mathsf{T}}Q_{i,\tau-1}x_{i,\tau-1} \\ & + u_{i,\tau-1}^{\mathsf{T}}\mathcal{R}_{i,\tau-1}u_{i,\tau-1} \\ & + (z_{i,\tau-1}^{k} - B_{i,\tau-1}^{\mathsf{T}}\mathcal{C}_{i,\tau})^{\mathsf{T}}u_{i,\tau-1} \\ \text{subject to}\ & x_{i,\tau} = A_{i,\tau-1}x_{i,\tau-1} + B_{i,\tau-1}u_{i,\tau-1}\end{aligned} \tag{49}$$

where

$$\mathcal{R}_{i,\tau} = R_{i,\tau} + \rho\sum_{j\in\mathcal{N}_i}\nu_{ij,\tau}^{i\mathsf{T}}\nu_{ij,\tau}^{i} \tag{50}$$

and

$$z_{i,\tau}^{k} = \nu_{i,\tau}^{i\mathsf{T}}q_{i,\tau}^{k} - 2\rho\sum_{j\in\mathcal{N}_i}\nu_{ij,\tau}^{i\mathsf{T}}\gamma_{ij,\tau}\left(u_{i,\tau}^{k}, u_{j,\tau}^{k}\right) \tag{51}$$

for $\tau \in \{1, \cdots, \mathcal{P}-1\}$. With

$$G_{i,\tau} = \left(B_{i,\tau}^{\mathsf{T}}\mathcal{H}_{i,\tau+1}B_{i,\tau} + \mathcal{R}_{i,\tau}\right)^{-1}, \tag{52}$$

agent i computes $\mathcal{H}_{i,\tau} \in \mathbb{R}^{n_i\times n_i}$ from the Riccati equation

$$\begin{aligned}\mathcal{H}_{i,\tau} = {} & A_{i,\tau}^{\mathsf{T}}\mathcal{H}_{i,\tau+1}A_{i,\tau} + Q_{i,\tau} \\ & - A_{i,\tau}^{\mathsf{T}}\mathcal{H}_{i,\tau+1}B_{i,\tau}G_{i,\tau}B_{i,\tau}^{\mathsf{T}}\mathcal{H}_{i,\tau+1}A_{i,\tau}\end{aligned} \tag{53}$$

beginning with $\mathcal{H}_{i,\mathcal{P}} = H_i$, for $\tau = 0, \cdots, \mathcal{P}-1$, and likewise, computes $\mathcal{C}_{i,\tau} \in \mathbb{R}^{n_i}$ from the Riccati equation

$$\mathcal{C}_{i,\tau} = A_{i,\tau}^{\mathsf{T}}\mathcal{C}_{i,\tau+1} + A_{i,\tau}^{\mathsf{T}}\mathcal{H}_{i,\tau+1}B_{i,\tau}G_{i,\tau}(z_{i,\tau}^{k} - B_{i,\tau}^{\mathsf{T}}\mathcal{C}_{i,\tau+1}) \tag{54}$$

beginning with $\mathcal{C}_{i,\mathcal{P}} = 0$.

By solving (49), agent i obtains its control inputs

$$u_{i,\tau}^{k+1} = -G_{i,\tau}\left(B_{i,\tau}^{\mathsf{T}}\mathcal{H}_{i,\tau+1}^{\mathsf{T}}A_{i,\tau}x_{i,\tau}^{k+1} + \frac{1}{2}(z_{i,\tau}^{k} - B_{i,\tau}^{\mathsf{T}}\mathcal{C}_{i,\tau+1})\right) \tag{55}$$

with the resulting value of the objective function given by

$$\begin{aligned}f_{i,\tau} = {} & x_{i,\tau}^{(k+1)\mathsf{T}}\mathcal{H}_{i,\tau}x_{i,\tau}^{(k+1)} - x_{i,\tau}^{(k+1)\mathsf{T}}\mathcal{C}_{i,\tau} \\ & - \frac{1}{4}\sum_{w=\tau}^{\mathcal{P}-1}\left(z_{i,w}^{k} - B_{i,w}^{\mathsf{T}}\mathcal{C}_{i,w+1}\right)^{\mathsf{T}}G_{i,w}\left(z_{i,w}^{k} - B_{i,w}^{\mathsf{T}}\mathcal{C}_{i,w+1}\right)\end{aligned} \tag{56}$$

for $\tau = 0, \cdots, \mathcal{P}-1$. Note that agent i needs to factorize the matrix in (52) only once when solving for its control inputs.

For each agent, the closed-loop control law in (55) consists of a feedback controller which depends on the current state of the agent and a feedforward controller which incorporates information from its neighbors into its inputs, enabling each agent to satisfy the coupling constraints in (39).

With (55), SOD-MPC provides a computationally efficient procedure for computing the optimal states and control inputs at every iteration. Agent i's state $x_{i,\tau}^{k+1}$ result from its dynamics model in (38) by applying $u_{i,\tau-1}^{k+1}$ at state $x_{i,\tau-1}^{k+1}$.

The closed-loop control law involves factorizing the matrix in (52). The factorization procedure can be performed using the Cholesky or LDL$^{\mathsf{T}}$ decomposition which requires $O(m_i^3)$ floating-point operations (FLOPS) in the dense case and $O(\mathrm{nnz}(L))$ FLOPS if sparsity is exploited, where $L \in \mathbb{R}^{m_i\times m_i}$ denotes a lower triangular matrix. Similarly, when solving (41) using a direct numerical approach, the Hessian of the Lagrangian of (41) can be factorized using the Cholesky or LDL$^{\mathsf{T}}$ decomposition, which would require $O(N_\tau^3(n_i^3 + m_i^3))$ FLOPS in the dense case and $O(\mathrm{nnz}(L))$ FLOPS if sparsity is exploited, where $L \in \mathbb{R}^{N_\tau(n_i+m_i)\times N_\tau(n_i+m_i)}$ denotes a lower triangular matrix. Hence, our approach reduces the computational complexity of solving (41) significantly, even after exploiting sparsity of the problem or reducing the problem size by eliminating the state variables and subsequently solving for the control inputs. Particularly, the computational complexity of our method does not depend on the length of the problem horizon, unlike the computational complexity of a direct numerical approach.

While other distributed model predictive control methods can be applied to solve (39), these methods do not consider the effects of time delays on the control inputs applied by each agent, which can be particularly significant in problems involving agents with fast dynamics. In contrast, SOD-MPC produces a closed-loop control law, providing robustness to time delays arising from the computation and communication procedures performed by each agent, in addition to modeling errors in the dynamics of the agents.

Algorithm 2 describes SOD-MPC for the convex model predictive control problem.

Algorithm 2: SOD-MPC for Closed-Loop Distributed Model Predictive Control

do in parallel $i = 1, \cdots, N$
- $(x_i, u_i) \leftarrow$ DistributedClosedControl$(t, \bar{x}_{i,t})$
- Get state $\bar{x}_{i,t'}$ at current time t'.
- $\breve{u}_{i,t'} \leftarrow u_{i,0}(\bar{x}_{i,t'})$
- Apply control input $\breve{u}_{i,t'}$.

while *task is in progress*;

The agents execute the *DistributedClosedControl* procedure to compute a closed-loop controller at each time t. On completion of this procedure, each agent measures its current state at time t' and computes its feedback control inputs at its current state, with its feedforward controller utilizing the most recent information received from its neighbors. Each agent repeats this procedure to obtain its control inputs at the next time instant. Noting that the states and control inputs obtained

Function DistributedClosedControl($t, \bar{x}_{i,t}$)

```
Initialization:
  k ← 0
  q_i^0 ← 0
  x_{i,0} ← x̄_{i,t}
do in parallel i = 1, ··· , N
  Compute H_{i,τ}, C_{i,τ} with (53), (54) ∀τ.
  for τ = 0, ··· , P − 1 do
    u_{i,τ}^{k+1} ← Equation (55)
    x_{i,τ+1}^{k+1} ← Equation (38)
  end
  Communication Step:
    Agent i shares the value of φ_{ij}^i(x_i^{k+1}, u_i^{k+1}) with
    agent j, its neighbor, and receives the value of
    φ_{ij}^j(x_j^{k+1}, u_j^{k+1}), ∀j ∈ N_i.
  q_i^{k+1} ← Equation (30)
  k ← k + 1
while not converged or stopping criterion is not met;
return (x_i, u_i)
```

using SOD-MPC always satisfy the dynamics constraints of each agent, the *DistributedClosedControl* procedure can be terminated early if needed. In these cases, each agent uses the most recent information received from its neighbors in computing its control inputs.

VII. Simulations

We examine the performance of SOD-MPC in convex and non-convex model predictive control problems, comparing our method to other distributed model predictive control methods including ADMM-MPC [31], [54] and dual decomposition [43]–[45], [48]. In ADMM-MPC, each agent optimizes over a copy of the states and control inputs of all its neighbors, with a consensus constraint between each pair of neighboring agents, ensuring the agents compute the same solution for equivalent variables. Consequently, ADMM-MPC differs from SOD-MPC, which does not require an agent to compute the states and control inputs of other agents. Rather, each agent in SOD-MPC optimizes over its contributions to the coupling constraints between the agent and its neighbors. We select the gradient ascent step-size in the dual decomposition algorithm and penalty parameters in ADMM-MPC and SOD-MPC using a modified binary search procedure, in which we examine the convergence rate of each algorithm at each candidate value of each parameter. Ultimately, we select the parameters that provide the fastest convergence rates in each method. We assume each agent represents the numerical values of its local optimization variables using the double precision floating-point format.

Remark 2. *ADMM-MPC does not consider the relevance of neighboring agents' variables to an agent's objective or constraint functions when assigning local variables to each agent. We modify ADMM-MPC to remove extraneous variables within each agent's set of local variables, obviating unnecessary optimization over these variables which would otherwise degrade the convergence rate of the method. In modifying ADMM-MPC, we further eliminate the need for computation of the dual variables at a central station, extending the method beyond problems with fully-connected communication networks.*

A. Convex Model Predictive Control

We consider a model predictive control problem involving agents with directional transceivers where optimum communication between agents requires alignment of the transceivers along an optimal surface (direction). For optimum communication while agents perform their desired tasks, we introduce a coupling constraint between neighbors, specifying the optimal surface for the transceivers for each pair of neighboring agents. The agents compute their optimal control inputs from the model predictive control problem

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{u}}{\text{minimize}} \ & \sum_{i=1}^{N} f_i(x_i, u_i) \\ \text{subject to } & x_{i,0} = \bar{x}_{i,t} \quad i = 1, \cdots, N \\ & x_{i,\tau+1} = A_{i,\tau} x_{i,\tau} + B_{i,\tau} u_{i,\tau} \quad \forall \tau,\ i = 1, \cdots, N \\ & \phi_{ij}(x_i, u_i, x_j, u_j) = 0 \quad \forall (i,j) \in \mathcal{E} \end{aligned} \tag{57}$$

where $\bar{x}_{i,t}$ represents the state of agent i at time t, $f_i(\cdot)$ represents the objective function in (40), and $\phi_{ij}(\cdot)$ describes the optimal surface for the transceivers on agents i and j. We define the coupling constraint between agents i and j as

$$\phi_{ij}(x_i, u_i, x_j, u_j) = e_{ij}^{\mathsf{T}}(x_i - x_j) \tag{58}$$

where $e_{ij} \in \mathbb{R}^{n_{ij}}$ specifies the normal to the desired hyperplane for optimal communication between agents i and j. By expressing the coupling constraint in (58) in the form given in (12), we obtain that

$$\phi_{ij}^i(x_i, u_i) = e_{ij}^{\mathsf{T}} x_i \quad \text{and} \quad \phi_{ij}^j(x_j, u_j) = e_{ij}^{\mathsf{T}} x_j. \tag{59}$$

With SOD-MPC, each agent only computes its states and control inputs without computing the states and control inputs of other agents, including its neighbors. Consequently, each agent does not communicate its control inputs to other agents. As such, in SOD-MPC, agent i communicates the value of $\phi_{ij}^i(\cdot)$, represented by 64 bits of information, with agent j. In contrast, each agent in ADMM-MPC shares 384 bits of information with its neighbors. Each agent in dual decomposition shares the same amount of information as agents in SOD-MPC.

We consider a randomly-generated connected communication network between the agents. In Figure 2, we show the transceivers on each agent along with the surfaces for optimal communication between a pair of transceivers. Each agent computes its optimal control inputs from (57) to arrive at its desired location, depicted in wireframe, while optimizing communication with its neighbors. By applying the resulting control inputs, the agents arrive at their desired locations, depicted in Figure 3.

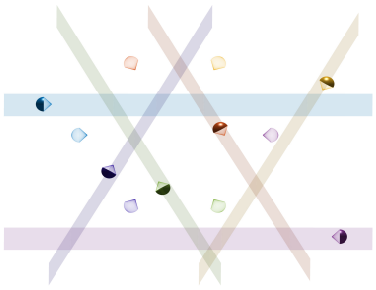

Fig. 2. Each agent has a directional transceiver displayed in the figure, with optimum communication between a pair of transceivers achieved when the transceivers lie on a collinear surface. By solving the model predictive control problem, each agent computes its control inputs to arrive at its desired location, depicted by the transceiver in wireframe, while optimizing communication with its neighbors.

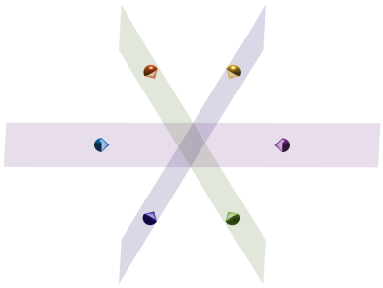

Fig. 3. At the end of the model predictive control problem, each agent arrives at its desired location with the transceivers oriented along collinear surfaces for optimum communication.

Convergence Analysis

We begin by examining the convergence rate of SOD-MPC to the optimal solution of the centralized problem compared to the convergence rate of ADMM-MPC and dual decomposition for the model predictive control problem in (57) with $N = 60$ agents and a sampling interval of 0.01 second. To visualize the solution obtained by each method in Figure 3, we set $n_i = 6$ and $m_i = 3$ for agent i. In certain problems, satisfaction of the coupling constraints between agents attain greater importance when these constraints influence the overall performance of all agents more significantly than the expended effort by each agent. As such, we compute the violations of the coupling constraints at each iteration along with the convergence of the value of the objective function to its optimal value.

Figure 4 shows the convergence rates of SOD-MPC, dual decomposition, and ADMM-MPC to the optimal solution. The objective value of the solution obtained using SOD-MPC converges faster to the optimal objective value compared to the other methods. SOD-MPC requires about 100 iterations for convergence to the optimal objective value compared to dual decomposition which converges in about 1000 iterations. ADMM-MPC exhibits the slowest convergence rate among all the methods. ADMM-MPC requires each agent to optimize over local copies of the optimization variables of other agents with which it shares a coupling constraint. As a result, agents in ADMM-MPC solve local optimization problems with a greater number of optimization variables compared to agents in SOD-MPC, leading to a slower convergence rate.

In addition, we examine the violations of the coupling constraints between agents in Figure 5. The violations of the coupling constraints between agents converge rapidly to zero with SOD-MPC, compared to dual decomposition and ADMM-MPC. Hence, SOD-MPC not only converges rapidly to the optimal objective value, the violations of the coupling constraints in SOD-MPC decreases rapidly likewise. As with convergence to the optimal objective value, the violations of the coupling constraints in SOD-MPC converge to zero in about 100 iterations while dual decomposition, which converges faster than ADMM-MPC, requires about 1000 iterations. With its faster convergence rates, SOD-MPC enables each agent to efficiently compute its control inputs, especially in problems with high control bandwidths. Moreover, faster convergence of the coupling constraints violations enables the application of SOD-MPC to problems where satisfaction of these constraints significantly influence the performance of all agents.

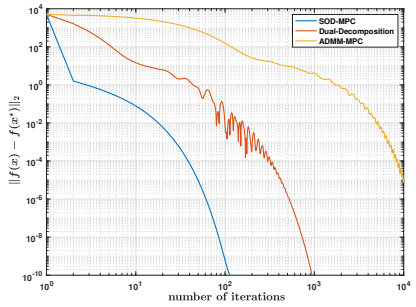

Fig. 4. Convergence of SOD-MPC, dual decomposition, and ADMM-MPC to the optimal objective value in convex model predictive control. SOD-MPC provides the fastest convergence rate to the optimal objective value, converging in about $\mathbf{100}$ iterations. Dual decomposition, which converges faster than ADMM-MPC, requires about $\mathbf{1000}$ iterations.

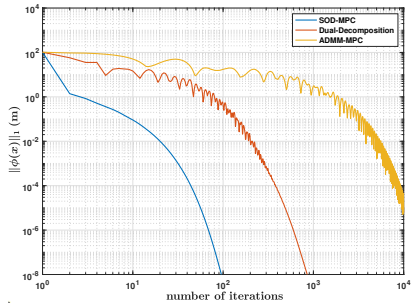

Fig. 5. Violation of the coupling constraints between agents using SOD-MPC, dual decomposition, and ADMM-MPC. The violations of these constraints in SOD-MPC converge rapidly to zero in about $\mathbf{100}$ iterations while dual decomposition requires about $\mathbf{1000}$ iterations for convergence of the coupling constraints violations.

Closed-Loop Performance

Regardless of the method applied in solving the model predictive control problem in (57), the computation and communication procedures performed by the agents or a central station introduce time delays between the specification of the model predictive control problem and the application of the control inputs by each agent. Note that the model

predictive control problem in (57) depends on the state of each agent at time t and other time-dependent problem parameters. Considering the time delays, the resulting control inputs will no longer be optimal for each agent at the time of application of the control inputs, which can severely impact the overall performance of all agents, especially in problems involving agents with fast dynamics.

We examine the performance of SOD-MPC, dual decomposition, ADMM-MPC, and an open-loop centralized method in the model predictive control problem (57) for different time delays in Figure 6. We do not modify the model predictive control problems to account for the time delay between the specification of the control problem and the application of the computed control inputs. In our simulations, we propagate each agent's dynamics forward in time assuming its control inputs are set to zero, while the agent is still computing its control inputs for the current instance of the model predictive optimization problem. At small time delays, all the methods attain almost the same mean violation of the coupling constraints between agents. However, the mean violation of the coupling constraints increases precipitously in dual decomposition, ADMM-MPC, and the open-loop centralized method as the duration of the time delay increases. These methods do not consider the effects of time delays in the controllers utilized by each agent; hence, the mean violation of the coupling constraints in the centralized method rises to almost 1000 m at a time delay of 100 ms. In contrast, SOD-MPC attains a mean coupling constraints violation close to 1 m even at a time delay of 100 ms. The closed-loop controller provided by SOD-MPC enables each agent to mitigate the effects of time delays, improving the performance of all agents in the model predictive control problem. Eventually, when the time delay exceeds 100 ms, the model predictive controllers resulting from dual decomposition, ADMM-MPC, and the open-loop centralized method become unstable.

In Figure 7, we show the violations of the coupling constraints for the first 60 time intervals at a time delay of 100 ms. The agents begin their tasks with their transceivers not aligned along the optimal surface for communication with their neighbors. Dual decomposition, ADMM-MPC, and the open-loop centralized method fail to align the transceivers on each agent to the optimal surface specified in the coupling constraints as the agents proceed with their tasks. As such, the violations of the coupling constraints do not converge to zero with these methods. However, the violations of the coupling constraints converge to zero in SOD-MPC, with each agent aligning its transceiver along the specified optimal surface in its coupling constraints.

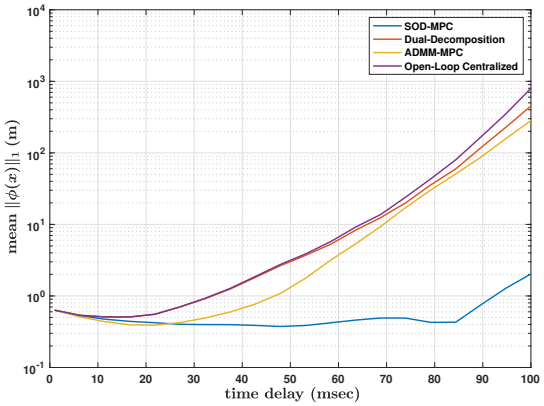

Fig. 6. Violations of the coupling constraints between agents for different time delays using SOD-MPC, dual decomposition, ADMM-MPC, and an open-loop centralized method. The mean violation of the coupling constraints increases rapidly for dual decomposition, ADMM-MPC, and the open-loop centralized method as the duration of the time delay increases. However, SOD-MPC exhibits robustness to time delays with a mean coupling constraints violation close to $\mathbf{1}$ m even at a time delay of $\mathbf{100}$ ms where the open-loop centralized method attains a mean coupling constraints violation close to $\mathbf{1000}$ m.

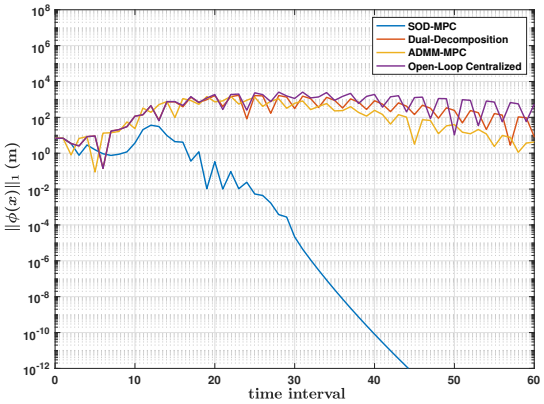

Fig. 7. Violations of the coupling constraints between agents for the first $\mathbf{60}$ time intervals at a time delay of $\mathbf{100}$ ms. While the violations of the coupling constraints do not converge with other methods, SOD-MPC attains convergence of the violations of the coupling constraints to zero.

B. Non-Convex Model Predictive Control

We consider the deployment of satellites using non-convex model predictive control. We assume the satellites are equipped with thrusters, enabling them to maneuver to specified desired positions and orientations. We represent the orientation of each satellite using a rotation matrix $C(t) \in \mathbb{R}^{3\times3}$ at time t with the angular velocity $\omega(t) \in \mathbb{R}^3$. Over a sampling interval δt with an angular velocity $\omega(t)$, the satellite rotates through an angle $\alpha \in \mathbb{R}$ around the axis $a \in \mathbb{R}^3$ with the dynamics of the rotation matrix given by the model

$$C(t+\delta t) = \mathcal{D}(t)C(t) \tag{60}$$

with

$$\mathcal{D}(t) = \cos(\alpha)I + (1-\cos(\alpha))aa^{\mathsf{T}} + \sin(\alpha)\hat{\omega} \tag{61}$$

where $I \in \mathbb{R}^{3\times3}$ represents the identity matrix, $\hat{\omega} \in \mathbb{R}^{3\times3}$ represents a skew-symmetric matrix derived from ω, $\alpha = \|\omega \cdot \delta t\|_2$, and $a = \frac{\omega\cdot\delta t}{\alpha}$.

We consider the presence of other orbiting agents and space debris in the vicinity of the satellites and thus introduce collision-avoidance constraints in the model predictive control problem to prevent collisions. In addition, we include non-convex coupling constraints between neighboring satellites which indicate the dependence of the quality of service provided by the group of satellites on the states of each satellite.

The satellites compute their inputs $\boldsymbol{\mathcal{F}}$ from the model predictive control problem

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{C},\boldsymbol{\mathcal{F}}}{\text{minimize}} \; & \sum_{i=1}^{N} f_i(x_i, C_i, \mathcal{F}_i) \\ \text{subject to } & d_i(x_i, C_i, \mathcal{F}_i) = 0 \quad i = 1, \cdots, N \\ & g_i(x_i, C_i, \mathcal{F}_i) = 0 \quad i = 1, \cdots, N \\ & h_i(x_i, C_i, \mathcal{F}_i) \leq 0 \quad i = 1, \cdots, N \\ & \phi_{ij}(x_i, C_i, \mathcal{F}_i, x_j, C_j, \mathcal{F}_j) = 0 \quad \forall (i,j) \in \mathcal{E} \end{aligned} \tag{62}$$

where $x_{i,\tau} \in \mathbb{R}^6$ represents the linear states and velocities of satellite i at time τ, $x_i = [x_{i,\tau}^{\mathsf{T}}, \tau = 0, \cdots, \mathcal{P}]^{\mathsf{T}}$ and $\boldsymbol{\mathcal{F}}$ and $\boldsymbol{\mathcal{C}}$ represent the concatenation of the inputs and rotation matrices of all the satellites respectively. We represent the dynamics constraints of satellite i with $d_i(\cdot)$ and include constraints on its initial and desired position, orientation, and velocities in $g_i(\cdot)$. In $h_i(\cdot)$, we specify the inputs constraints as well as the collision-avoidance constraints of satellite i, given by

$$\text{dist}(x_i, p) \geq \lambda_{\min} \quad \forall p \in \mathcal{A} \tag{63}$$

where $\mathcal{A}$ comprises of the state of all space debris detected by satellite i and the state of all satellites in its vicinity. We denote the minimum distance between a satellite and other agents as $\lambda_{\min}$. In this problem, we define the distance function $\text{dist}(\cdot)$ using the Euclidean norm. We specify that neighboring satellites maintain the same altitude, giving rise to the non-convex coupling constraint $\phi_{ij}(\cdot)$ between satellites i and j.

Using SOD-MPC, satellite i computes its inputs $\mathcal{F}_i \in \mathbb{R}^6$ locally, without computing the states and inputs of other satellites. Each satellite applies its resulting inputs, satisfying its coupling constraints with other satellites while avoiding collisions with other orbiting agents and space debris, as depicted in Figure 8. At each iteration, each agent in SOD-MPC and dual decomposition shares 64 bits of information with its neighbors. In contrast, each agent in ADMM-MPC shares 384 bits of information with its neighbors.

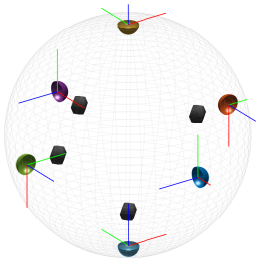

Fig. 8. Deployment of satellites to their orbits in space using non-convex model predictive control. Using SOD-MPC, each satellite computes its control inputs locally while satisfying coupling constraints with other satellites and avoiding collisions with space debris, denoted by the gray objects.

We assess the performance of SOD-MPC, dual decomposition, and ADMM-MPC in this problem (62), over a randomly-generated connected communication network with $N = 12$ satellites and a 0.01 second sampling interval, solving each problem using the interior-point optimization solver IPOPT [67] with the linear solver MA-57 and the maximum number of iterations at 300. We examine the convergence of the value of the objective function obtained from each method to the objective value from a centralized method in Figure 9. The objective value of the solution obtained using SOD-MPC converges faster than the objective value resulting from dual decomposition and ADMM-MPC, converging in about 600 iterations to the optimal objective value. ADMM-MPC converges faster than dual decomposition, requiring about 1000 iterations for convergence.

Likewise, we examine the convergence of the violations of the coupling constraints between the satellites in (62) to zero for SOD-MPC, dual decomposition, and ADMM-MPC. SOD-MPC attains the fastest convergence rate with the violations of the coupling constraints converging to zero in about 600 iterations. ADMM-MPC requires about 1000 iterations for convergence of the coupling constraints violations to zero, converging faster than dual decomposition. Hence, SOD-MPC provides a more efficient method for computing the control inputs of each satellite in addition to satisfying the coupling constraints between the satellites.

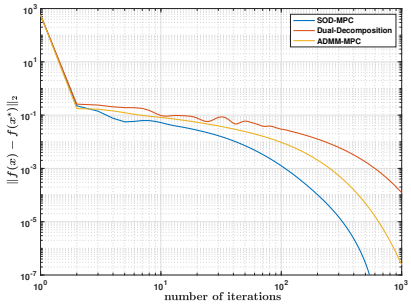

Fig. 9. Convergence of the objective value obtained using SOD-MPC, dual decomposition, and ADMM-MPC to the objective value obtained from a centralized method. SOD-MPC provides the fastest convergence rate and converges in about **600** iterations. ADMM-MPC, which converges faster than dual decomposition, requires about **1000** iterations for convergence.

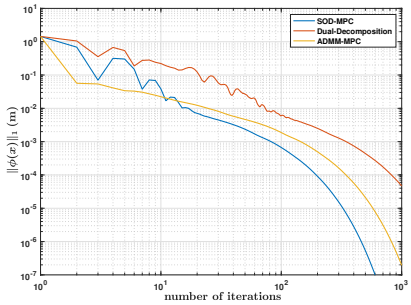

Fig. 10. Convergence of the violations of the coupling constraints between agents using SOD-MPC, dual decomposition, and ADMM-MPC. SOD-MPC converges faster than dual decomposition and ADMM-MPC in about **600** iterations, with ADMM-MPC converging faster than dual decomposition in about **1000** iterations.

VIII. Conclusion

We introduce an algorithm for distributed model predictive control, SOD-MPC, where each agent computes its control inputs locally, in parallel, without computing the the control

inputs of other agents irrelevant to its performance. Each agent does not rely on a central station for computing its control inputs, unlike some other distributed methods. Our algorithm achieves linear convergence to the optimal control inputs, providing faster convergence rates compared to other distributed model predictive control methods including dual decomposition, in convex problems with strongly convex objective functions. In addition, SOD-MPC produces a closed-loop controller in convex model predictive control problems with affine constraints, providing robustness to time delays arising from the communication and computation procedures performed by the agents.

IX. Appendix

A. Proof of Lemma 1

We provide the proof of Lemma 1. For the problem in (31), we express the coupling constraints between all agents as

$$\Phi(\boldsymbol{u}) = \mathcal{J}\boldsymbol{u} + \eta \tag{64}$$

where $\mathcal{J}$ represents the Jacobian of the coupling constraints with respect to $\boldsymbol{u}$ and η represents constant terms in the coupling constraints. For convex problems, note that the coupling constraints are affine. As such, we can always obtain the constraint in (64) in any convex model predictive control problem. We can further express the Jacobian as

$$\mathcal{J} = \vec{\mathcal{J}} - \overleftarrow{\mathcal{J}} \tag{65}$$

where $\vec{\mathcal{J}}_{ij}$ denotes the Jacobian of the coupling constraint $\phi_{ij}(\cdot)$, represented by the edge $(i,j) \in \mathcal{E}$, with $\phi^j_{ij}(\cdot) = 0$ and $\overleftarrow{\mathcal{J}}_{ij}$ denotes the Jacobian of $\phi_{ij}(\cdot)$ with $\phi^i_{ij}(\cdot) = 0$. Likewise, we express the constant terms in the coupling constraints as

$$\eta = \vec{\eta} - \overleftarrow{\eta} \tag{66}$$

with $\vec{\eta}_{ij} = \eta^i_{ij}$ and $\overleftarrow{\eta}_{ij} = \eta^j_{ij}$. By defining $\bar{\mathcal{J}}$ and $\bar{\eta}$ as

$$\begin{aligned} \bar{\mathcal{J}} &= \vec{\mathcal{J}} + \overleftarrow{\mathcal{J}} \\ \bar{\eta} &= \vec{\eta} + \overleftarrow{\eta}\ , \end{aligned} \tag{67}$$

we can express the update procedures for the auxiliary variables c and dual variables q of all agents (30) as

$$c^{k+1} = \frac{1}{2}\left(\bar{\mathcal{J}}\boldsymbol{u}^{k+1} + \bar{\eta}\right) \tag{68}$$

with

$$q^{k+1} = q^k + \rho\left(\mathcal{J}\boldsymbol{u}^{k+1} + \eta\right). \tag{69}$$

We define the W-norm as $\|\cdot\|_W$ with

$$W = \begin{bmatrix} \frac{1}{4\rho}I & 0 \\ 0 & \rho I \end{bmatrix} \tag{70}$$

where $W > 0$.

Following the procedure in (26), the control inputs of all agents satisfy

$$\nabla\mathbb{F}\left(\boldsymbol{u}^{k+1}\right) + \frac{1}{2}\mathcal{J}^{\mathsf{T}}q^{k+1} + \rho\bar{\mathcal{J}}^{\mathsf{T}}\left(c^{k+1} - c^k\right) = 0 \tag{71}$$

for the update procedure at iteration k. The optimal control inputs satisfy the optimality conditions

$$\nabla\mathbb{F}\left(\boldsymbol{u}^{\star}\right) + \frac{1}{2}\mathcal{J}^{\mathsf{T}}q^{\star} = 0 \tag{72}$$

with the optimal auxiliary variables and dual variables satisfying

$$\begin{aligned} c^{\star} &= \frac{1}{2}\left(\bar{\mathcal{J}}\boldsymbol{u}^{\star} + \bar{\eta}\right) \\ 0 &= \rho\left(\mathcal{J}\boldsymbol{u}^{\star} + \eta\right) \end{aligned} \tag{73}$$

from (68) and (69). With (71) and (72), we obtain

$$\begin{aligned} \nabla\mathbb{F}\left(\boldsymbol{u}^{k+1}\right) - \nabla\mathbb{F}\left(\boldsymbol{u}^{\star}\right) + \frac{1}{2}\mathcal{J}^{\mathsf{T}}\left(q^{k+1} - q^{\star}\right) \\ + \rho\bar{\mathcal{J}}^{\mathsf{T}}\left(c^{k+1} - c^k\right) = 0\ , \end{aligned} \tag{74}$$

along with

$$\begin{aligned} c^{k+1} - c^{\star} &= \frac{1}{2}\bar{\mathcal{J}}\left(\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right) \\ q^{k+1} - q^k &= \rho\mathcal{J}\left(\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right) \end{aligned} \tag{75}$$

from (68), (69), and (73).

For a strongly convex function $f(x)$,

$$\left(\nabla f(x_a) - \nabla f(x_b)\right)^{\mathsf{T}}(x_a - x_b) \geq m_f\|x_a - x_b\|_2^2 \tag{76}$$

where $m_f > 0$. For the convex model predictive control problem in (31),

$$\begin{aligned} &\left(\nabla\mathbb{F}\left(\boldsymbol{u}^{k+1}\right) - \nabla\mathbb{F}\left(\boldsymbol{u}^{\star}\right)\right)^{\mathsf{T}}\left(\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right) \\ &= -\left(\frac{1}{2}\mathcal{J}^{\mathsf{T}}\left(q^{k+1} - q^{\star}\right) + \rho\bar{\mathcal{J}}^{\mathsf{T}}\left(c^{k+1} - c^k\right)\right)^{\mathsf{T}}\left(\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right) \\ &= -2\left(z^{k+1} - z^{\star}\right)^{\mathsf{T}}W\left(z^{k+1} - z^k\right) \\ &= \left\|z^k - z^{\star}\right\|_W^2 - \left\|z^{k+1} - z^{\star}\right\|_W^2 - \left\|z^{k+1} - z^k\right\|_W^2 \end{aligned} \tag{77}$$

from (75). Consequently,

$$\begin{aligned} &\left\|z^k - z^{\star}\right\|_W^2 - \left\|z^{k+1} - z^{\star}\right\|_W^2 \\ &\qquad \geq m_f\left\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right\|_2^2 + \left\|z^{k+1} - z^k\right\|_W^2 \end{aligned} \tag{78}$$

from (76).

B. Proof of Lemma 2

We prove Lemma 2. With Lipschitz continuity of $\nabla\mathbb{F}(\boldsymbol{u})$,

$$\left\|\nabla\mathbb{F}(\boldsymbol{u}^{k+1}) - \nabla\mathbb{F}(\boldsymbol{u}^{\star})\right\|_2^2 \leq L_f^2\left\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right\|_2^2 \tag{79}$$

where $L_f > 0$. From (68),

$$\left\|c^{k+1} - c^{\star}\right\|_2^2 \leq \frac{1}{4}\left\|\bar{\mathcal{J}}\right\|_2^2\left\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right\|_2^2. \tag{80}$$

Using the relation $\|a+b\|_2^2 \geq (\mu-1)\|a\|_2^2 + \frac{\mu-1}{\mu}\|b\|_2^2$ for $\mu > 1$ and (74),

$$\begin{aligned} &\left\|\frac{1}{2}\mathcal{J}^{\mathsf{T}}\left(q^{k+1} - q^{\star}\right) + \rho\bar{\mathcal{J}}^{\mathsf{T}}\left(c^{k+1} - c^k\right)\right\|_2^2 \\ &\qquad \geq \frac{\mu-1}{4\mu}\sigma^2_{\min}\left(\mathcal{J}\right)\left\|q^{k+1} - q^{\star}\right\|_2^2 \\ &\qquad\quad + (1-\mu)\rho^2\sigma^2_{\min}\left(\bar{\mathcal{J}}\right)\left\|c^{k+1} - c^k\right\|_2^2 \end{aligned} \tag{81}$$

which gives

$$\begin{aligned} &\frac{\mu-1}{4\mu}\sigma^2_{\min}\left(\mathcal{J}\right)\left\|q^{k+1} - q^{\star}\right\|_2^2 \\ &+ (1-\mu)\rho^2\sigma^2_{\min}\left(\bar{\mathcal{J}}\right)\left\|c^{k+1} - c^k\right\|_2^2 \\ &\qquad \leq L_f^2\left\|\boldsymbol{u}^{k+1} - \boldsymbol{u}^{\star}\right\|_2^2\ , \end{aligned} \tag{82}$$

resulting in

$$\frac{1}{4\rho}\left\|q^{k+1}-q^{\star}\right\|_2^2+\rho\left\|c^{k+1}-c^{\star}\right\|_2^2 \leq \frac{\rho\mu\sigma_{\min}^2\left(\bar{\mathcal{J}}\right)}{\sigma_{\min}^2\left(\mathcal{J}\right)}\left\|c^{k+1}-c^{k}\right\|_2^2 + \left(\frac{\mu L_f^2}{\rho(\mu-1)\sigma_{\min}^2\left(\mathcal{J}\right)}+\frac{\rho}{4}\sigma_{\max}^2\left(\bar{\mathcal{J}}\right)\right)\left\|\boldsymbol{u}^{k+1}-\boldsymbol{u}^{\star}\right\|_2^2 \tag{83}$$

with (80).

Simplifying (83) gives

$$\frac{\ell}{4\rho}\left\|q^{k+1}-q^{\star}\right\|_2^2+\ell\rho\left\|c^{k+1}-c^{\star}\right\|_2^2 \leq \rho\left\|c^{k+1}-c^{k}\right\|_2^2+m_f\left\|\boldsymbol{u}^{k+1}-\boldsymbol{u}^{\star}\right\|_2^2 \tag{84}$$

where

$$\ell=\min\left\{\frac{4m_f\rho(\mu-1)\sigma_{\min}^2\left(\mathcal{J}\right)}{\rho^2(\mu-1)\sigma_{\max}^2\left(\bar{\mathcal{J}}\right)\sigma_{\min}^2\left(\mathcal{J}\right)+4\mu L_f^2},\frac{\sigma_{\min}^2\left(\mathcal{J}\right)}{\mu\sigma_{\min}^2\left(\bar{\mathcal{J}}\right)}\right\} \tag{85}$$

with $\ell>0$. From (78) and (84),

$$\ell\left\|z^{k+1}-z^{\star}\right\|_W^2\leq\left\|z^{k}-z^{\star}\right\|_W^2-\left\|z^{k+1}-z^{\star}\right\|_W^2\ , \tag{86}$$

resulting in

$$\frac{\left\|z^{k+1}-z^{\star}\right\|_W^2}{\left\|z^{k}-z^{\star}\right\|_W^2}\leq\frac{1}{\ell+1}\ , \tag{87}$$

showing Q-linear convergence of $\{z^k\}$ to $z^{\star}$.

References

[1] M. Mercangöz and F. J. Doyle III, "Distributed model predictive control of an experimental four-tank system," *Journal of process control*, vol. 17, no. 3, pp. 297–308, 2007.

[2] D. B. Pourkargar, A. Almansoori, and P. Daoutidis, "Impact of decomposition on distributed model predictive control: A process network case study," *Industrial & Engineering Chemistry Research*, vol. 56, no. 34, pp. 9606–9616, 2017.

[3] T. Faulwasser, L. Grüne, M. A. Müller *et al.*, "Economic nonlinear model predictive control," *Foundations and Trends® in Systems and Control*, vol. 5, no. 1, pp. 1–98, 2018.

[4] B. Stellato, T. Geyer, and P. J. Goulart, "High-speed finite control set model predictive control for power electronics," *IEEE Transactions on power electronics*, vol. 32, no. 5, pp. 4007–4020, 2016.

[5] T. Dragičević, "Model predictive control of power converters for robust and fast operation of ac microgrids," *IEEE Transactions on Power Electronics*, vol. 33, no. 7, pp. 6304–6317, 2017.

[6] G. Mantovani and L. Ferrarini, "Temperature control of a commercial building with model predictive control techniques," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 4, pp. 2651–2660, 2014.

[7] G. Gholamibozanjani, J. Tarragona, A. De Gracia, C. Fernández, L. F. Cabeza, and M. M. Farid, "Model predictive control strategy applied to different types of building for space heating," *Applied energy*, vol. 231, pp. 959–971, 2018.

[8] J. Álvarez, J. Redondo, E. Camponogara, J. Normey-Rico, M. Berenguel, and P. Ortigosa, "Optimizing building comfort temperature regulation via model predictive control," *Energy and Buildings*, vol. 57, pp. 361–372, 2013.

[9] D. Picard, J. Drgoňa, M. Kvasnica, and L. Helsen, "Impact of the controller model complexity on model predictive control performance for buildings," *Energy and Buildings*, vol. 152, pp. 739–751, 2017.

[10] M. Maasoumy, M. Razmara, M. Shahbakhti, and A. S. Vincentelli, "Handling model uncertainty in model predictive control for energy efficient buildings," *Energy and Buildings*, vol. 77, pp. 377–392, 2014.

[11] G. Bianchini, M. Casini, A. Vicino, and D. Zarrilli, "Demand-response in building heating systems: A model predictive control approach," *Applied Energy*, vol. 168, pp. 159–170, 2016.

[12] Y. Kwak, J.-H. Huh, and C. Jang, "Development of a model predictive control framework through real-time building energy management system data," *Applied Energy*, vol. 155, pp. 1–13, 2015.

[13] R. Tang and S. Wang, "Model predictive control for thermal energy storage and thermal comfort optimization of building demand response in smart grids," *Applied Energy*, vol. 242, pp. 873–882, 2019.

[14] Y. Zheng, S. E. Li, K. Li, F. Borrelli, and J. K. Hedrick, "Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 899–910, 2016.

[15] E. van Nunen, J. Reinders, E. Semsar-Kazerooni, and N. Van De Wouw, "String stable model predictive cooperative adaptive cruise control for heterogeneous platoons," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 186–196, 2019.

[16] J. Wang, S. Gong, S. Peeta, and L. Lu, "A real-time deployable model predictive control-based cooperative platooning approach for connected and autonomous vehicles," *Transportation Research Part B: Methodological*, vol. 128, pp. 271–301, 2019.

[17] K. Yu, H. Yang, X. Tan, T. Kawabe, Y. Guo, Q. Liang, Z. Fu, and Z. Zheng, "Model predictive control for hybrid electric vehicle platooning using slope information," *IEEE Transactions on intelligent transportation systems*, vol. 17, no. 7, pp. 1894–1909, 2016.

[18] R. Kianfar, P. Falcone, and J. Fredriksson, "A control matching model predictive control approach to string stable vehicle platooning," *Control Engineering Practice*, vol. 45, pp. 163–173, 2015.

[19] L. Chisci, T. Pecorella, and R. Fantacci, "Dynamic bandwidth allocation in geo satellite networks: a predictive control approach," *Control Engineering Practice*, vol. 14, no. 9, pp. 1057–1067, 2006.

[20] K. Xiao, S. Mao, and J. K. Tugnait, "Maq: A multiple model predictive congestion control scheme for cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 4, pp. 2614–2626, 2017.

[21] E. Henriksson, D. E. Quevedo, E. G. Peters, H. Sandberg, and K. H. Johansson, "Multiple-loop self-triggered model predictive control for network scheduling and control," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 6, pp. 2167–2181, 2015.

[22] M. Neunert, C. De Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear model predictive control for unified trajectory optimization and tracking," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 1398–1404.

[23] C. Shen and Y. Shi, "Distributed implementation of nonlinear model predictive control for auv trajectory tracking," *Automatica*, vol. 115, p. 108863, 2020.

[24] M. Bangura and R. Mahony, "Real-time model predictive control for quadrotors," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 11 773–11 780, 2014.

[25] A. Alessandretti, A. P. Aguiar, and C. N. Jones, "Trajectory-tracking and path-following controllers for constrained underactuated vehicles using model predictive control," in *2013 european control conference (ecc)*. IEEE, 2013, pp. 1371–1376.

[26] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 2, pp. 952–964, 2016.

[27] P. D. Christofides, R. Scattolini, D. M. de la Pena, and J. Liu, "Distributed model predictive control: A tutorial review and future research directions," *Computers & Chemical Engineering*, vol. 51, pp. 21–41, 2013.

[28] M. Farina and R. Scattolini, "Distributed predictive control: A non-cooperative algorithm with neighbor-to-neighbor communication for linear systems," *Automatica*, vol. 48, no. 6, pp. 1088–1096, 2012.

[29] A. Bestler and K. Graichen, "Distributed model predictive control for continuous-time nonlinear systems based on suboptimal admm," *Optimal Control Applications and Methods*, vol. 40, no. 1, pp. 1–23, 2019.

[30] H. Zheng, R. R. Negenborn, and G. Lodewijks, "Fast admm for distributed model predictive control of cooperative waterborne agvs," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1406–1413, 2016.

[31] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Püschel, "Distributed admm for model predictive control and congestion control," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 5110–5115.

[32] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia, "Cooperative distributed model predictive control," *Systems & Control Letters*, vol. 59, no. 8, pp. 460–469, 2010.

[33] M. A. Müller, M. Reble, and F. Allgöwer, "Cooperative control of dynamically decoupled systems via distributed model predictive control,"

International Journal of Robust and Nonlinear Control, vol. 22, no. 12, pp. 1376–1397, 2012.

[34] O. Shorinwa, T. Halsted, and M. Schwager, "Scalable distributed optimization with separable variables in multi-agent networks," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 3619–3626.

[35] B. T. Stewart, S. J. Wright, and J. B. Rawlings, "Cooperative distributed model predictive control for nonlinear systems," *Journal of Process Control*, vol. 21, no. 5, pp. 698–704, 2011.

[36] M. Heidarinejad, J. Liu, D. M. de la Peña, J. F. Davis, and P. D. Christofides, "Multirate lyapunov-based distributed model predictive control of nonlinear uncertain systems," *Journal of Process Control*, vol. 21, no. 9, pp. 1231–1242, 2011.

[37] J. Liu, D. Muñoz de la Peña, and P. D. Christofides, "Distributed model predictive control of nonlinear process systems," *AIChE journal*, vol. 55, no. 5, pp. 1171–1184, 2009.

[38] J. Liu, D. M. de la Peña, and P. D. Christofides, "Distributed model predictive control of nonlinear systems subject to asynchronous and delayed measurements," *Automatica*, vol. 46, no. 1, pp. 52–61, 2010.

[39] M. R. Jovanovic and B. Bamieh, "Lyapunov-based distributed control of systems on lattices," *IEEE Transactions on Automatic Control*, vol. 50, no. 4, pp. 422–433, 2005.

[40] A. N. Venkat, J. B. Rawlings, and S. J. Wright, "Stability and optimality of distributed model predictive control," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 6680–6685.

[41] B. Alrifaee, F.-J. Heßeler, and D. Abel, "Coordinated non-cooperative distributed model predictive control for decoupled systems using graphs," *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 216–221, 2016.

[42] S. Riverso and G. Ferrari-Trecate, "Tube-based distributed control of linear constrained systems," *Automatica*, vol. 48, no. 11, pp. 2860–2865, 2012.

[43] Y. Wakasa, M. Arakawa, K. Tanaka, and T. Akashi, "Decentralized model predictive control via dual decomposition," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 381–386.

[44] F. Farokhi, I. Shames, and K. H. Johansson, "Distributed mpc via dual decomposition and alternative direction method of multipliers," in *Distributed model predictive control made easy*. Springer, 2014, pp. 115–131.

[45] A. Grancharova and T. A. Johansen, "Distributed quasi-nonlinear model predictive control by dual decomposition," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 1429–1434, 2011.

[46] C. Conte, C. N. Jones, M. Morari, and M. N. Zeilinger, "Distributed synthesis and stability of cooperative distributed model predictive control for linear systems," *Automatica*, vol. 69, pp. 117–125, 2016.

[47] J. Koko, "A survey on dual decomposition methods," *SeMA Journal*, vol. 62, no. 1, pp. 27–59, 2013.

[48] P. Giselsson, M. D. Doan, T. Keviczky, B. De Schutter, and A. Rantzer, "Accelerated gradient methods and dual decomposition in distributed model predictive control," *Automatica*, vol. 49, no. 3, pp. 829–833, 2013.

[49] S. Koehler, C. Danielson, and F. Borrelli, "A primal-dual active-set method for distributed model predictive control," *Optimal Control Applications and Methods*, vol. 38, no. 3, pp. 399–419, 2017.

[50] E. Camponogara and H. F. Scherer, "Distributed optimization for model predictive control of linear dynamic networks with control-input and output constraints," *IEEE Transactions on Automation Science and Engineering*, vol. 8, no. 1, pp. 233–242, 2010.

[51] 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.

[52] Z. Wang and C. J. Ong, "Distributed model predictive control of linear discrete-time systems with local and global constraints," *Automatica*, vol. 81, pp. 184–195, 2017.

[53] R. Rostami, G. Costantini, and D. Görges, "Admm-based distributed model predictive control: Primal and dual approaches," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 6598–6603.

[54] T. H. Summers and J. Lygeros, "Distributed model predictive consensus via the alternating direction method of multipliers," in *2012 50th annual Allerton conference on communication, control, and computing (Allerton)*. IEEE, 2012, pp. 79–84.

[55] P. Braun, T. Faulwasser, L. Grüne, C. M. Kellett, S. R. Weller, and K. Worthmann, "Hierarchical distributed admm for predictive control with applications in power networks," *IFAC Journal of Systems and Control*, vol. 3, pp. 10–22, 2018.

[56] X. Hou, Y. Xiao, J. Cai, J. Hu, and J. E. Braun, "Distributed model predictive control via proximal jacobian admm for building control applications," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 37–43.

[57] P. Giselsson and A. Rantzer, "On feasibility, stability and performance in distributed model predictive control," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 1031–1036, 2013.

[58] J. Köhler, M. A. Müller, and F. Allgöwer, "Distributed model predictive control—recursive feasibility under inexact dual optimization," *Automatica*, vol. 102, pp. 1–9, 2019.

[59] P. O. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Transactions on Automatic Control*, vol. 44, no. 3, pp. 648–654, 1999.

[60] G. Pannocchia, J. B. Rawlings, and S. J. Wright, "Conditions under which suboptimal nonlinear mpc is inherently robust," *Systems & Control Letters*, vol. 60, no. 9, pp. 747–755, 2011.

[61] V. Kekatos and G. B. Giannakis, "Distributed robust power system state estimation," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1617–1626, 2012.

[62] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Püschel, "Distributed optimization with local domains: Applications in mpc and network flows," *IEEE Transactions on Automatic Control*, vol. 60, no. 7, pp. 2004–2009, 2014.

[63] 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.

[64] L. Chen, D. Sun, and K.-C. Toh, "A note on the convergence of admm for linearly constrained convex optimization problems," *Computational Optimization and Applications*, vol. 66, no. 2, pp. 327–343, 2017.

[65] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (admm): quadratic problems," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 644–658, 2014.

[66] A. U. Raghunathan and S. Di Cairano, "Optimal step-size selection in alternating direction method of multipliers for convex quadratic programs and model predictive control," in *Proceedings of symposium on mathematical theory of networks and systems*. Citeseer, 2014, pp. 807–814.

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