Distributed Nonlinear Trajectory Optimization for Multi-Robot Motion Planning

Laura Ferranti, Lorenzo Lyons, Rudy R. Negenborn, Tamás Keviczky, *Senior Member, IEEE*, and Javier Alonso-Mora, *Senior Member, IEEE*

***Abstract*— This work presents a method for multi-robot coordination based on a novel distributed nonlinear model predictive control (NMPC) formulation for trajectory optimization and its modified version to mitigate the effects of packet losses and delays in the communication among the robots. Our algorithms consider that each robot is equipped with an onboard computation unit to solve a local control problem and communicate with neighboring autonomous robots via a wireless network. The difference between the two proposed methods is in the way the robots exchange information to coordinate. The information exchange can occur in a following: 1) synchronous or 2) asynchronous fashion. By relying on the theory of the nonconvex alternating direction method of multipliers (ADMM), we show that the proposed solutions converge to a (local) solution of the centralized problem. For both algorithms, the communication exchange preserves the safety of the robots; that is, collisions with neighboring autonomous robots are prevented. The proposed approaches can be applied to various multi-robot scenarios and robot models. In this work, we assess our methods, both in simulation and with experiments, for the coordination of a team of autonomous vehicles in the following: 1) an unsupervised intersection crossing and 2) the platooning scenarios.**

***Index Terms*— Collision avoidance, fault-tolerant control, multi-robot systems, optimal control, optimization.**

I. Introduction

EVERY year, thousands of people are involved in transportation-related accidents with fatal consequences [1], [2]. A key component to address this issue and reduce the amount of fatalities in the current transportation and mobility network is the development of *connected and automated mobility* solutions [3], [4]. These autonomous vehicles will soon be part of our daily life, transporting goods, or people to their destinations. One of the main challenges is that of generating collision-free trajectories that coordinate these traffic participants, to ensure the safety of the vehicles and of the humans. Formally, this is a multi-robot coordination problem. Through the use of a central coordinator or by relying on the communication among neighboring robots, these algorithms should require only a small amount of information (e.g., current pose and speed of the neighbors) to decide on a safe navigation strategy. In addition, while distributed algorithms are more resilient to faults in the communication strategy (compared with the centralized approaches that have one single point of failure), these algorithms strongly rely on the communication among the robots. Hence, they are still vulnerable to communication faults (e.g., packet loss and delays) that can compromise the overall safety of the coordination scheme.

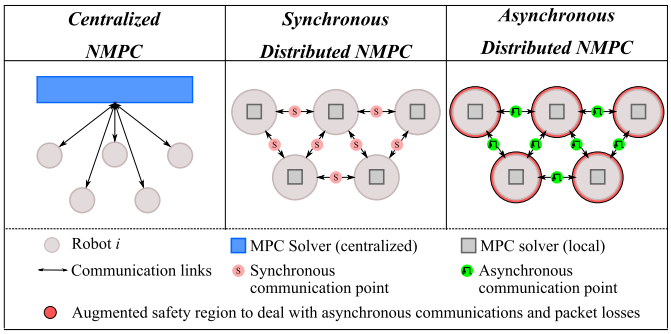


Fig. 1. Overview of the proposed approaches for multi-robot coordination based on the use of distributed NMPC.

This article presents two methods to solve a *centralized* multi-robot coordination problem in a *distributed* fashion, that is, without the need of a coordinator.

1) A *synchronous algorithm*, that is, a distributed nonconvex trajectory optimization method to coordinate multiple robots with guarantees on the convergence to the solution of the original *centralized* coordination problem.
2) An *asynchronous algorithm*, that is, a modified version of the aforementioned distributed algorithm to account for a more realistic communication network. The algorithm is able to deal with non-blocking exchanges of information among the robots, communication delays, and packet losses, while retaining guarantees on the convergence to a suboptimal solution of the original *centralized* problem.

Fig. 1 summarizes the approaches we propose, highlighting the differences with the original centralized approach (in which a

Manuscript received 31 March 2022; accepted 12 September 2022. The work of Laura Ferranti was supported by the Dutch Science Foundation Dutch Research Council (NWO)-Applied and Engineering Science (TTW) within the Veni Project HARMONIA under Grant 18165. Recommended by Associate Editor A. Girard. *(Corresponding author: Laura Ferranti.)*

The authors are with the Faculty of Mechanical, Maritime, and Materials Engineering, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: l.ferranti@tudelft.nl; l.lyons@tudelft.nl; r.r.negenborn@tudelft.nl; t.keviczky@tudelft.nl; j.alonsomora@tudelft.nl).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCST.2022.3211130.

Digital Object Identifier 10.1109/TCST.2022.3211130

central coordinator computes a feasible path for all the robots) and in the way the robots communicate.

We evaluate our methods for the coordination of a team of autonomous vehicles in different scenarios (that would usually be treated with different tailored solutions), namely, an unsupervised intersection crossing scenario and a platoon formation scenario. Finally, we present a practical implementation of the method using a team of small-scale autonomous cars.

Much research in mobile–robot motion planning focuses on safety, that is, how to avoid collisions with other robots and with the environment (e.g., road boundaries and lanes). Classical methods for coordination in complex dynamic environments use reactive strategies [5], [6], [7], [8], [9], [10], assume a priority order [11], or rely on scheduling [12], [13], [14], [15], [16] to coordinate robots. These methods do not explicitly consider the interactions between robots and moving obstacles. Learning-based methods [17], [18], [19], [20], [21] can be used to consider these interactions at the cost of losing interpretability. Constrained-optimization approaches [22], [23], [24], [25] can be used to consider interactions without losing interpretability, but at potentially high computational cost. Our proposed framework for multi-robot coordination fits in this last category, and it aims at reducing the computational load using the decomposition methods.

Several authors proposed centralized optimization-based approaches for multi-robot coordination and intersection negotiation ([26], [27], [28], [29] to mention a few). In contrast, our algorithms do not require a central coordinator and are distributed via a communication channel. Several authors also provided distributed solutions. De Campos [30] proposed a decentralized design in which the robots compute their decisions sequentially. The approach is based on scheduling, invariant sets, and optimization-based techniques. Jiang et al. [31] proposed a parallelizable approach under a given precedence order. Collisions among the robots are avoided by sharing the arrival and departure times at the intersection. In [32], an extension of [31] is provided to handle rear-end collisions. Our design does not require a precedence order and does not require a discretization of the environment, operating directly in continuous space. Katriniok et al. [33] proposed a distributed model predictive control (MPC) design that relies on constraint prioritization and uses semidefinite programming relaxations to deal with the nonconvexity of the collision avoidance constraints. Borrelli et al. [34] and Keviczky et al. [35] rely on a decentralized linear MPC formulation for collision-free formation control. Zheng et al. [36], [38], Chen et al. [37], and Rey et al. [39] proposed distributed algorithms that rely on the alternating direction method of multipliers (ADMM). Chen et al. [37] used distributed linearized MPC to coordinate autonomous vessels in the presence of environmental disturbances. In [38], the distributed MPC problems are used for the cooperative multivessel system. Compared with the aforementioned approaches, we rely on nonlinear MPC (NMPC); that is, we do not linearize the system dynamics or the collision avoidance constraints, but we consider directly the nonlinear dynamics and nonconvex constraints in the problem formulation to reduce the conservatism in the behavior of the robots. Furthermore, we provide asymptotic convergence guarantees by relying on the theory of nonlinear ADMM (NADMM) [40]. The solver presented in [40] cannot be trivially applied to our coordination problem. NADMM only handles linear coupling constraints among the robots, while our coordination problem introduces a nonlinear coupling caused by the presence of nonconvex collision avoidance constraints. We reformulate our coordination problem in an appropriate form that both preserve the solution of the original coordination problem, and are suitable for NADMM. We achieve this goal by introducing the following: 1) a new set of decision variables that act as shared variables among the robots and 2) a new set of constraints that handle the *consensus* among the robots. The resulting optimization problem is the sum of smaller (one for each robot) linearly coupled subproblems that can be solved using NADMM. We show that the proposed synchronous algorithm converges to a locally optimal solution (due to the nonconvex nature of the problem) of the coordination problem (see Theorem 1).

Our approach relies on MPC. In recent years, MPC has gained attention in applications with fast dynamics, such as automotive [26], [41], [42], [43], waterborne transport [23], [44], [45], and aeronautics [46], [47], because of great improvements in terms of solvers used for online optimization [48], [49], [50], [51], [52]. MPC is also a promising technique for fault-tolerant control. Several authors showed the ability of MPC to deal with actuator faults [47], [53], [54], [55]. Less attention has been given to the potential of MPC to deal with communication faults. Izadi et al. [56] proposed a decentralized convex MPC design to deal with delays for a leader–follower formation control problem. Izadi et al. [57] proposed a distributed path following control strategy to explicitly account for time-varying communication delays based on event-triggered communications. Pena and Christofides [58] proposed a distributed strategy for the regulation problem of nonlinear systems subject to data losses-based using a Lyapunov-based MPC formulation. Compared with the previous approaches, our coordination strategies strongly rely on the NADMM framework to shape the interactions among the robots. In addition, we tailored the distributed coordination strategy to mitigate packet losses and delays by leveraging the MPC ability of generating predictions over a finite time horizon. Within the ADMM literature, some strategies have been proposed to deal with delays and packet losses (e.g., [59], [60]). Compared with these approaches, we exploit the features of the MPC design to provide predictions, and we use predictive optimization techniques to compensate for the packet loss and delays in the ADMM framework. In addition, we rely on the NADMM [40] strategy able to deal with the sum of nonconvex functions. The solver proposed in [40] and our synchronous algorithm, however, require the robots to exchange information at given synchronization points. This assumption can be unrealistic for practical robotics applications, in which the communication among the robots can be affected by delays or packet loss. This article proposes a tailored version of the NADMM solver (our asynchronous algorithm) that leverages the ability of NMPC to provide predictions to compensate for imperfect behavior of the solver due to communication faults.

We build on [42], which relies on the model predictive contouring control (MPCC) formulation proposed in [61] and [62], to design the local trajectory generation strategies. Compared with [42], we look at the distributed multiagent planning and coordination problem. We focus on the algorithms required to find a solution for such a problem using NMPC combined with NADMM to distribute the problem.

We presented a preliminary version of the synchronous algorithm in the conference paper [23]. Compared with [23], we completely revise the NADMM strategy and provide theoretical guarantees for the synchronous algorithm. Furthermore, we design the asynchronous algorithm to deal with a more realistic communication framework among the robots. Finally, we extend and update our simulation results with a comparison with the centralized version of the local motion planner applied to the control of a team of autonomous vehicles.

This article is structured as follows. Section II provides preliminary information to make the article self-contained, including the NADMM strategy presented in [40] and the centralized NMPC problem formulation. Sections III and IV describe the synchronous and asynchronous designs, respectively. Section V presents an evaluation of the proposed methods. Finally, Section VI concludes this article.

II. Preliminaries

A. Nonlinear ADMM

Given that our approach relies on the NADMM presented in [40], this section provides some useful definitions and a short overview of NADMM. The functions introduced in this section are meant to explain the generic version of NADMM.

In this article, all the vectors are indicated with a bold symbol. The two norm of a vector $\mathbf{u}$ is $\|\mathbf{u}\|$. Let $\bar{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$ and $\mathbf{Range}(A)$ indicate the extended-real line and the range (column space) of a matrix $A \in \mathbb{R}^{m\times n}$, respectively. Furthermore, let $\mathrm{eig}_{\min}(A)$ indicate the minimum eigenvalue of a matrix $A \in \mathbb{R}^{n\times n}$. Finally, let $\mathbf{dom}\, f$ be the domain of a function $f : \mathbb{R}^n \to \bar{\mathbb{R}}$. The following definitions are also useful.

Definition 1 [40]: Given $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ and a linear operator $A \in \mathbb{R}^{m\times m}$, the image function $(Af) : \mathbb{R}^m \to [-\infty, +\infty]$ is defined as $(Af)(\boldsymbol{\sigma}) := \inf_{\mathbf{x}\in\mathbb{R}^n}\{f(\mathbf{x})|A\mathbf{x} = \boldsymbol{\sigma}\}$.

Definition 2 [63]: A function $f(\mathbf{x})$ is lower semicontinuous at $\bar{\mathbf{x}}$ if $\liminf_{\mathbf{x}\to\bar{\mathbf{x}}} f(\mathbf{x}) = f(\bar{\mathbf{x}})$.

Consider the generic problem below

$$\min_{\boldsymbol{\xi}\in\mathbb{R}^{n_{\boldsymbol{\xi}}},\mathbf{y}\in\mathbb{R}^{n_{\mathbf{y}}}} J_{\boldsymbol{\xi}}(\boldsymbol{\xi}) + J_{\mathbf{y}}(\mathbf{y}) \tag{1a}$$

$$\text{s.t. } A\boldsymbol{\xi} + B\mathbf{y} = \mathbf{b} \tag{1b}$$

where $J_{\boldsymbol{\xi}}(\boldsymbol{\xi}) : \mathbb{R}^{n_{\boldsymbol{\xi}}} \to \mathbb{R}$ and $J_{\mathbf{y}}(\mathbf{y}) : \mathbb{R}^{n_{\mathbf{y}}} \to \bar{\mathbb{R}}$ are the proper and lower semicontinuous functions of $\boldsymbol{\xi}$ and $\mathbf{y}$, respectively. These cost functions can be nonconvex. $A \in \mathbb{R}^{n_{\mathfrak{b}}\times n_{\boldsymbol{\xi}}}$, $B \in \mathbb{R}^{n_{\mathfrak{b}}\times n_{\mathbf{y}}}$, and $\mathbf{b} \in R^{n_{\mathfrak{b}}}$ define the linear coupling constraints (1b).

The augmented Lagrangian associated with Problem (1) is defined as follows:

$$\mathcal{L}_\rho(\boldsymbol{\xi}, \mathbf{y}, \boldsymbol{\lambda}) := J_{\boldsymbol{\xi}}(\boldsymbol{\xi}) + J_{\mathbf{y}}(\mathbf{y}) + \langle \boldsymbol{\lambda}, A\boldsymbol{\xi} + B\mathbf{y} - \mathbf{b}\rangle \tag{2a}$$

$$+ \frac{\rho}{2}\|A\boldsymbol{\xi} + B\mathbf{y} - \mathbf{b}\|^2 \tag{2b}$$

where $\rho > 0$ is a penalty parameter, and $\boldsymbol{\lambda} \in \mathbb{R}^{n_{\mathfrak{b}}}$ is the Lagrange multiplier associated with constraint (1b).

To solve Problem (1), the NADMM algorithm iteratively solves the following steps [40]:

$$\begin{cases} \boldsymbol{\lambda}^{+/2} & = \boldsymbol{\lambda} - \rho(1-\beta)(A\boldsymbol{\xi} + B\mathbf{y} - \mathbf{b}) \\ \boldsymbol{\xi}^{+} & \in \operatorname{argmin} \mathcal{L}_\rho\left(\cdot, \mathbf{y}, \boldsymbol{\lambda}^{+/2}\right) \\ \boldsymbol{\lambda}^{+} & = \boldsymbol{\lambda}^{+/2} + \rho(A\boldsymbol{\xi}^{+} + B\mathbf{y} - \mathbf{b}) \\ \mathbf{z}^{+} & \in \operatorname{argmin} \mathcal{L}_\rho(\boldsymbol{\xi}^{+}, \cdot, \boldsymbol{\lambda}^{+}) \end{cases}$$

where $\beta \in (0, 2)$ is a tuning parameter. NADMM is an iterative algorithm that converges to a (locally) optimal solution of Problem (1) only asymptotically.

Section III details how to reformulate the multi-robot coordination problem to fit in the structure of Problem (1) [see Problem (13)] and solve it in a distributed fashion (see Algorithms 1 and 2).

B. Model of the Robots

We consider autonomous robots whose dynamics can be represented by the following nonlinear discrete-time model:

$$\mathbf{x}_i(t+1) = f_i(\mathbf{x}_i(t), \mathbf{u}_i(t)), \quad i \in \mathcal{I}_V \tag{3}$$

where $\mathbf{x}_i(t) \in \mathbb{R}^{n_i}$ represents the state of Robot i, $\mathbf{u}_i(t) \in \mathbb{R}^{m_i}$ represents the associated control command, $f_i : \mathbb{R}^{n_i} \times \mathbb{R}^{m_i} \to \mathbb{R}^{n_i}$ represents the (possibly) nonlinear dynamics of Robot i, V is the number of robots, and $\mathcal{I}_V := \{1, \ldots, V\}$.

The following assumptions hold throughout this article.

Assumption 1: The robots communicate only within a communication radius c_r; that is, there exists a communication link between Robots i and j if and only if Robot i is in the communication radius of Robot j. In practice, the robots will use the information concerning the neighboring robots only when they are within the planning horizon of the local motion planner.

Assumption 2: All the robots within the communication radius are fully autonomous and communicate with the neighboring robots (i.e., we do not consider mixed traffic scenarios, in which we have noncommunicating robots).

The model description above is general and can be employed in different multi-robot applications (e.g., for the coordination of autonomous cars, ships, drones, and aircraft). In case of continuous-time systems, the model above can be obtained by discretization of the continuous-time dynamics, as illustrated in Section V.

C. Collision Avoidance Constraints

This section introduces the strategy used to represent each robot and formulate the collision avoidance constraints in the coordination problem. We use a strategy similar to the one proposed in [42] for autonomous cars.

Without loss of generality, we explain the representation for two robots, namely, Robots i and j.[1] Fig. 2 depicts the

[1]We note that the method applies in a straightforward way to the case of $V \geq 2$ robots by including additional constraints for each of the other robots analogously to the case for Robot j (as depicted in Fig. 2, where we added a third robot, namely, Robot k, to represent a more general scenario).

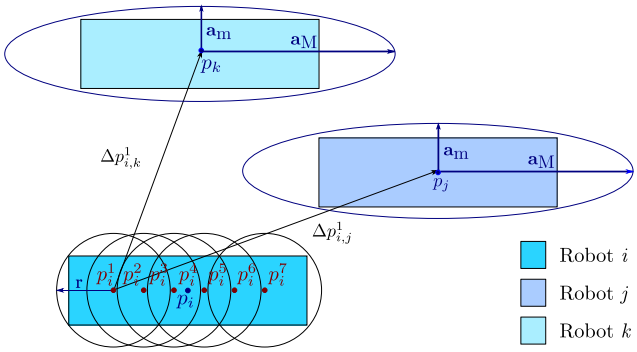


Fig. 2. Robots' representation for collision avoidance from the perspective of Robot i: the circles represent Robot i, while the ellipses describe its neighbors.

proposed approach from the perspective of Robot i. Robot i is represented as n_disk disks of radius r centered in $\mathbf{p}_i^h$ (where we used $\mathbf{p}$ to indicate the position on the (x, y) plane in the body frame, $h \in \mathcal{I}^\mathrm{disk} := \{1, 2, \ldots, n_\mathrm{disk}\}$, and n_disk is the number of disks used to describe the robot). From the perspective of Robot i, Robot j is represented as an ellipse with semimajor axis a_M (longitudinal direction) and a_m (lateral direction), respectively.

Collision avoidance is achieved when the disks used to represent Robot i and the ellipses representing the neighboring robots do not intersect. As proposed in [42], using an approximation of the Minkowski sum[2] between each disk representing Robot i and the ellipse representing the neighboring robot, the collision avoidance constraints between Robot i and the neighbor j have the following analytical representation:

$$\underbrace{\big(R(\eta_j)\big(\mathbf{p}_i^h - \mathbf{p}_j\big)\big)^\mathrm{T} \begin{bmatrix} \dfrac{1}{(a_M+r)^2} & 0 \\ 0 & \dfrac{1}{(a_m+r)^2} \end{bmatrix} R(\eta_j)\big(\mathbf{p}_i^h - \mathbf{p}_j\big) > 1}_{\mathbf{c}_{i,j}^h} \tag{4}$$

where $R(\eta_j)$ is the rotation matrix of Robot j (η_j is the orientation of Robot j according to the reference frame indicated in Fig. 2). The constraints above require the pose of Robot j, that is, $\mathbf{p}_j$ and $\boldsymbol{\eta}_j$. In the following, we assume a homogeneous team of robots (i.e., the same a_m, a_M, and r) to simplify the notation. Nevertheless, the approach can be easily extended to a heterogeneous team of robots, by adding as additional shared information a_{M_j}, a_{m_j}, and r_i.

D. Problem Formulation

Our approach relies on MPC. The MPC controller repeatedly solves an optimization problem based on the available plant measurements to compute the optimal sequence of control commands over a finite time window, called the prediction horizon. Only the first control command of this sequence is applied to the plant in closed loop in a receding-horizon fashion [64].

[2]An over-approximation can be achieved following the approach proposed in [25].

Following [42] and extending it to the context of multi-robot coordination, we formulate a trajectory-generation problem in which the controller aims to minimize the error with respect to a path dependent on the path parameter ϕ, rather than a time-dependent reference signal. This formulation allows the controller flexibility in the definition of the state variables (they are not constrained to follow a time-dependent trajectory). Furthermore, this formulation allows the controller to select a desired reference velocity without compromising the choice of the reference path.

The goal of the proposed coordination algorithm is to minimize the *sum* of the following costs[3]:

$$J_i := q_{v_x} \left\| v_i^\mathrm{ref} - v_{x_i} \right\|^2 + \mathbf{e}_i^\mathrm{T} Q_e \mathbf{e}_i \tag{5}$$

where v_{x_i} is the velocity in the longitudinal direction, v_i^ref is the desired speed, q_{v_x} and Q_e are tuning parameters, and the error $\mathbf{e}_i \in \mathbb{R}^2$ for Robot i is defined as follows:

$$\mathbf{e}_i := \begin{bmatrix} e_i^l & e_i^c \end{bmatrix}^\mathrm{T}. \tag{6}$$

The quantities e_i^l and e_i^c are the *longitudinal error* (i.e., the error in the path's tangential direction) and the *contouring error* (i.e., the error in the path's normal direction), respectively. At time step t, the longitudinal error is defined as follows:

$$e_i^l := -[\cos\eta_i(\phi_i) \ \sin\eta_i(\phi_i)]\big(\mathbf{p}_i - \mathbf{p}_i^\mathrm{ref}\big) \tag{7}$$

where $\eta_i(\phi_i)$, $\mathbf{p}_i$, and $\mathbf{p}_i^\mathrm{ref}$ are the heading of the path, the position on the (x, y)-plane of Robot i, and the reference path on the (x, y)-plane, respectively. Similarly, at time step t, the contouring error is defined as follows:

$$e_i^c := [\sin\eta_i(\phi_i) \ -\cos\eta_i(\phi_i)]\big(\mathbf{p}_i - \mathbf{p}_i^\mathrm{ref}\big). \tag{8}$$

More details on the derivation of the longitudinal and contouring errors can be found in [42].

At time t, our goal is to solve the following trajectory optimization problem:

$$\min_{\mathbf{x}_i, \mathbf{u}_i, \boldsymbol{\phi}_i, i \in \mathcal{I}_V} \sum_{i=1}^{V} \Bigg(\sum_{k=0}^{N-1} J_i(\mathbf{x}_i(t+k), \mathbf{u}_i(t+k)) + J_i(\mathbf{x}_i(t+N)) \Bigg) \tag{9a}$$

$$\text{s.t.:} \ \ \mathbf{x}_i(t+k+1) = f_i(\mathbf{x}_i(t+k), \mathbf{u}_i(t+k)) \tag{9b}$$

$$\mathbf{x}_i(t) = \mathbf{x}_i^\mathrm{init} \tag{9c}$$

$$G_i(\mathbf{x}_i(t+k), \quad \mathbf{u}_i(t+k)) \le \mathbf{g}_i \tag{9d}$$

$$\mathbf{c}_{i,j}^h(t+k) > 1, \quad j \neq i, \quad h \in \mathcal{I}^\mathrm{disk} \tag{9e}$$

where constraints (9b)–(9e) are for $i = 1, \ldots, V$ and $k = 1, \ldots, N_i$. In Problem (9), $\mathbf{x}_i$ and $\mathbf{u}_i$ represent the predicted evolution of the state and control command of Robot i, respectively, over the prediction horizon N. The vector $\mathbf{x}_i^\mathrm{init} \in \mathbb{R}^{n_i}$ represents the current measured state of the robot. Furthermore, (9d) indicates the constraints on the states and actuators of the robots (with G_i and $\mathbf{g}_i$ constant matrices of appropriate dimensions). In addition, (9e) represents the collision avoidance constraints (4) between Robots i and j ($j \neq i$) along the prediction horizon N. These constraints

[3]To simplify the notation, we omit the time and robot dependency when it is clear from the context.

are present only if the neighboring robots are within the communication radius c_r.

The goal of each robot is to minimize the local J_i (9a). The navigation must comply with the following: 1) the dynamics of each robot [expressed by the *dynamic constraints* (9b)] and 2) the physical constraints on the state and control command of each robot [expressed by constraint (9d)]. Furthermore, the navigation must comply with safety requirements of collision avoidance with the other robots moving in the same area (expressed by the nonconvex constraints (9e) detailed in Section II-C).

Assumption 3: We assume that the coordination problem (9) has a feasible solution.

Problem (9) requires a central coordinator to compute the appropriate control command for all the robots in the network (this processing unit can be required, for example, to handle an intersection). First, the central coordinator has to solve the predictive control problem online (i.e., within the sampling time of the fastest robot). This can cause problems for the scalability of the proposed approach, when the number of robots increases. Second, having a central coordinator means that all the robots must be willing to share information concerning their dynamics, constraints, and objectives with the central node. This might be problematic for vehicle manufacturers, which may not be open to share information concerning their products. To efficiently solve Problem (9), we need to remove the requirement of having a central coordinator by allowing the robots to communicate in a tailored manner with each other.

III. Synchronous Distributed NMPC

In the following, Section III-A describes our proposed reformulation of Problem (9). Then, Section III-B shows how to decompose the problem to solve it using NADMM. Finally, Section III-C provides convergence guarantees when using the proposed decomposition to coordinate the robots.

A. Reformulation of Problem (9)

In Problem (9), the *only* coupling among the different robots is represented by the collision avoidance constraints (9e). Each robot needs a local copy of the predicted position and orientation along the prediction horizon of its neighboring robots to solve its local optimization problem. A simple strategy could be that each robot computes its control command using the predicted position and orientation that its neighbors computed at the previous time instant (i.e., no ADMM). This strategy can be suitable when the robots' global paths are not in conflict (e.g., independent highway lanes). When the robots need to reach *consensus* on a safe path (e.g., to cross an intersection or change multiple lanes), relying only on previously computed paths can be unsafe. In these scenarios, tailored information exchanges can guarantee collision-free trajectories (if a feasible solution of the centralized problem exists), as discussed later in the section.

We propose the following reformulation in which each Robot i ($i = 1, \ldots, V$) solves the following problem:

$$\min_{\mathbf{z}_i, \mathbf{p}_i^h} \sum_{k=0}^{N} J_i(\mathbf{z}_i(t+k)) \tag{10a}$$

$$\text{s.t.: (9b)–(9d)} \tag{10b}$$

$$\mathbf{p}_{j|i}(t+k) = \mathbf{p}_i^h(t+k) - \Delta\mathbf{p}_{i,j}^h(t+k), \quad j \neq i, \ h \in \mathcal{I}^{\text{disk}} \tag{10c}$$

$$\mathbf{p}_i(t+k) = \bar{\mathbf{p}}_j^h(t+k) - \Delta\mathbf{p}_{j,i}^h(t+k), \quad j \neq i, \ h \in \mathcal{I}^{\text{disk}} \tag{10d}$$

$$\eta_i(t+k) - \eta_{i|j}(t+k) = 0 \tag{10e}$$

$$\mathbf{p}_i^h(t+k) = R^h(\mathbf{z}_i(t+k))\mathbf{z}_i(t+k), \quad h \in \mathcal{I}^{\text{disk}} \tag{10f}$$

$$\mathbf{c}_{(i,j)|i}^h(t+k) > 1, \quad j \neq i, \ h \in \mathcal{I}^{\text{disk}} \tag{10g}$$

$$\mathbf{c}_{(j,i)|i}^h(t+k) > 1, \quad j \neq i, \ h \in \mathcal{I}^{\text{disk}} \tag{10h}$$

where constraints (10c)–(10h) are for $k = 1, \ldots, N$ and $\mathbf{z}_i := [\mathbf{x}_i^{\mathrm{T}}, \mathbf{u}_i^{\mathrm{T}}]^{\mathrm{T}}$ (to simplify the notation, we omit the fact that the input is applied from 0 to $N-1$). Note that in the problem formulation above, $\mathbf{z}_i$ and $\mathbf{p}_i^h$ are *local* variables (i.e., variables whose values are computed on board of Robot i), $\mathbf{p}_{j|i}$ is the local information that Robot i has of Robot j's position, and $\mathbf{p}_{i|j}$ is the local information that Robot j has of Robot i's position. Similarly, $\eta_{i|j}$ is the local information that Robot j has concerning the orientation of Robot i. In addition, the values of $\Delta\mathbf{p}_{i,j}^h$ are newly introduced consensus variables that carry the information of the distance between each disk representing Robot i and Robot j. Similarly, the values of $\Delta\mathbf{p}_{j,i}^h$ are consensus variables carrying the information of the distance between each disk representing Robot j and Robot i. Furthermore, we modify the notation ($c_{(i,j)|i}^h$ instead of $c_{i,j}^h$) for the collision avoidance constraints in (4) to indicate that $\mathbf{c}_{(i,j)|i}^h$ uses $\mathbf{p}_{j|i}$ instead of $\mathbf{p}_j$ ($\eta_{j|i}$ instead of η_j), and that $\mathbf{c}_{(j,i)|i}^h$ uses $\mathbf{p}_{j|i}$ instead of $\mathbf{p}_j$. In the remainder of this article, we use

$$\boldsymbol{\xi}_i := \left[\mathbf{z}_i^{\mathrm{T}} \left(\mathbf{p}_i^1\right)^{\mathrm{T}}, \ldots, \left(\mathbf{p}_i^h\right)^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{n_\xi}$$

as the vector of local variables.

Compared with Problem (9), the local problems above contain additional constraints, namely, constraints (10c)–(10e). In these constraints, the newly introduced vectors $\Delta\mathbf{p}_{i,j}^h$, $\Delta\mathbf{p}_{j,i}^h$, and $\eta_{j|i}$ are used to break up the coupling between Robot i and its neighbors caused by the collision avoidance constraints. The introduction of these variables (and of the associated constraints) is fundamental to create a linear coupling among the robots and reformulate the trajectory optimization problem in the standard NADMM form (1). Finally, constraint (10f) indicates the nonlinear relationship [highlighted by the matrix $R^h(\mathbf{z}_i(t+k))$] between the center of the disks describing each robot and the center of the robot itself.

Fig. 3 highlights the local decision variables, copies, and collision avoidance constraints of each robot (Robots i and j). Furthermore, the figure highlights the global variables that our design uses to ensure consensus among the robots.

The introduction of the nonlinear local equality constraints (10f) and a new set of local variables might seem redundant. Their introduction, however, is fundamental for NADMM, which requires linear coupling constraints [40]. Without them, the constraints in (11) would be nonlinear in the decision variables, and the convergence results proposed in [40] would not hold. Recently, promising ADMM versions with nonconvex coupling constraints have been proposed

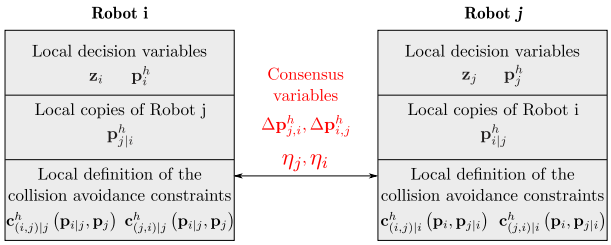


Fig. 3. Summary of the local and shared information after the introduction of the consensus constraints.

(e.g., [65]), but the convergence results are still limited and cannot be employed for our application.

We rewrite in a more compact notation constraints (10c)–(10d) as follows:

$$A_i\boldsymbol{\xi}_i(t+k)+B_i\mathbf{y}_i(t+k)=\mathbf{b}_i(t+k),\quad k=1,\dots,N \tag{11}$$

where

$$\mathbf{y}_i:=\left[\Delta\mathbf{p}_{i,j}^{1^{\mathrm{T}}},\dots,\Delta\mathbf{p}_{i,j}^{h^{\mathrm{T}}}\Delta\mathbf{p}_{j,i}^{1^{\mathrm{T}}},\dots,\Delta\mathbf{p}_{j,i}^{h^{\mathrm{T}}}\right]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathbf{y}}} \tag{12a}$$

$$A_i:=\left[E_i^T,\dots,E_h^{\mathrm{T}}\,F,\dots,F\right]^{\mathrm{T}}\in\mathbb{R}^{n_A\times n_{\boldsymbol{\xi}}} \tag{12b}$$

$$B_i:=[-I,\dots,-I\;I,\dots,I]^{\mathrm{T}}\in\mathbb{R}^{n_A\times n_{\mathbf{y}}} \tag{12c}$$

$$\mathbf{b}_i^{\mathrm{T}}:=\left[(\mathbf{p}_{j|i})^T,\dots,(\mathbf{p}_{j|i})^T\,(\mathbf{p}_{j|i}^1)^{\mathrm{T}},\dots,(\mathbf{p}_{j|i}^h)^T\right]\in\mathbb{R}^{n_A} \tag{12d}$$

where E_i selects the components of $\mathbf{p}_i^h$ from the vector of local variables $\boldsymbol{\xi}_i$, and F selects the components of $\mathbf{p}_i$ from $\boldsymbol{\xi}_i$. We can rewrite Problem (10) in a more compact notation as follows:

$$\min_{\boldsymbol{\xi}_i,\mathbf{y}_i}\ \sum_{k=1}^{N}J_i(\boldsymbol{\xi}_i(t+k)) \tag{13a}$$

$$\text{s.t.:}\ \ \boldsymbol{\xi}_i\in\mathcal{F}_i \tag{13b}$$

$$A_i\boldsymbol{\xi}_i(t+k)+B_i\mathbf{y}_i(t+k)=\mathbf{b}_i(t+k) \tag{13c}$$

where $\mathcal{F}_i:=\{\mathbf{z}_i|$ (9b)–(9d), (10f)–(10h) are satisfied$\}$ is the feasible region of Robot i.

By taking the sum of Problem (13) for all the neighboring robots, we recover the original MPC formulation (9). For example, if we consider (10c) for two neighboring robots (namely, robots i and j), the following hold:

$$(i):\ \Delta\mathbf{p}_{i,j}^h(t+k)=\mathbf{p}_i^h(t+k)-\mathbf{p}_{j|i}(t+k),\quad j\neq i,\ \ h\in\mathcal{I}^{\text{disk}} \tag{14a}$$

$$(j):\ \Delta\mathbf{p}_{i,j}^h(t+k)=\mathbf{p}_{i|j}^h(t+k)-\mathbf{p}_j(t+k),\quad j\neq i,\ \ h\in\mathcal{I}^{\text{disk}}. \tag{14b}$$

If both (14a) and (14b) are satisfied, it means that $\mathbf{p}_{i|j}=\mathbf{p}_i$ and $\mathbf{p}_{j|i}=\mathbf{p}_j$. Hence, Problem (13) [and, consequently, Problem (10)] is a reformulation of Problem (9).

The fundamental difference between the two problems [i.e., Problems (9) and (13)] is that the sum of the local problems (13) can be solved in a distributed fashion by relying on the NADMM solver [40].

B. Problem Decomposition Using NADMM

Algorithm 1 shows the proposed control strategy that relies on NADMM (Steps 5–17) [40]. Neighboring robots have to communicate to evaluate and exchange the locally computed values[4] of $\Delta\mathbf{p}_{i,j}^h$ (Steps 11 and 15). Furthermore, compared with the strategy proposed in [40], we have to deal with constraints in the inner problems solved by the local robots (Step 9). Each robot needs an optimizer able to solve the local constrained nonconvex problems (such as FORCES Pro [51]). In this respect, the augmented Lagrangian associated with Problem (13) for all the robots is defined as follows:

$$\mathcal{L}(\boldsymbol{\xi},\mathbf{y},\boldsymbol{\lambda}):=J(\boldsymbol{\xi})+\langle\boldsymbol{\lambda},A\boldsymbol{\xi}+B\mathbf{y}-\mathbf{b}\rangle+(\rho/2)\|A\boldsymbol{\xi}+B\mathbf{y}-\mathbf{b}\|^2$$

where $J:=\sum_{i=0}^{V}J_i$, $\boldsymbol{\xi}:=[\boldsymbol{\xi}_0^{\mathrm{T}},\dots,\boldsymbol{\xi}_V^{\mathrm{T}}]^{\mathrm{T}}\in\mathbb{R}^{n_{\boldsymbol{\xi}}(V)}$, $\mathbf{y}:=[\mathbf{y}_0^{\mathrm{T}},\dots,\mathbf{y}_N^{\mathrm{T}}]^{\mathrm{T}}\in\mathbb{R}^{n_{\mathbf{y}}(V)}$, $\mathbf{b}:=[\mathbf{b}_0^{\mathrm{T}},\dots,\mathbf{b}_V^{\mathrm{T}}]^{\mathrm{T}}\in\mathbb{R}^{n_A(V)}$, and A and B are defined accordingly from (11). Notice the similarity with (2). The update of the global variables is performed in Step 14 of Algorithm 1 as follows:

$$\Delta\mathbf{p}_{i,j}^h:=\frac{(\mathbf{p}_i^h-\mathbf{p}_{j|i})+(\mathbf{p}_{i|j}^h-\mathbf{p}_j)}{2},\quad \eta_{j|i}:=\eta_j. \tag{15}$$

Their values are obtained by taking the values of $\Delta\mathbf{p}_{i,j}^h$ computed according to the mean of the measurements available to Robots i and j, respectively, at iteration s of the solver for all h. Note that to perform the update above, it is sufficient for Robot j to communicate the difference $\mathbf{p}_{i|j}^h-\mathbf{p}_j$ and η_j along the prediction horizon for all $h\in\mathcal{I}^{\text{disk}}$ values.

At time 0, each robot will need to initialize $\mathbf{y}_i^0$. We assume that the first time a robot enters in the communication radius of another robot, it will communicate the required current information to initialize the first element of the vector (which is a parameter in the local optimization problems). Then, the ADMM steps will be used to optimize the consensus variables online.

The vector $\mathbf{b}_i$ varies along the prediction horizon, but it is not a decision variable. We can precompute its values as follows. Robot i receives/updates iteratively the values of

Algorithm 1 Synchronous Distributed NMPC

1: **for** $i=1,\dots,V$ **do**
2: Given $\boldsymbol{\xi}_i^0$, $\mathbf{y}_i^0$, $\boldsymbol{\lambda}_i^0$, ρ, $\beta\in(0,2]$.
3: **end for**
4: **for** $t=0,1,2,\dots$ **do**
5: **for** $\mathrm{s}=1,\dots,\mathtt{iter}_{\max}$ **do**
6: **for** $i=1,\dots,V$ each robot computes in parallel **do**
7: Update $\mathbf{b}_i$.
8: $\hat{\boldsymbol{\lambda}}_i^s\leftarrow\boldsymbol{\lambda}_i^s-\rho(1-\beta)(A_i\boldsymbol{\xi}_i^s+B_i\mathbf{y}_i^s-\mathbf{b}_i)$.
9: $\boldsymbol{\xi}_i^{s+1}\leftarrow\operatorname{argmin}_{\boldsymbol{\xi}_i\in\mathcal{F}_i}\mathcal{L}_i\left(\boldsymbol{\xi}_i,\mathbf{y}_i^s,\hat{\boldsymbol{\lambda}}_i^s\right)$.
10: $\boldsymbol{\lambda}_i^{s+1}\leftarrow\hat{\boldsymbol{\lambda}}_i^s+\rho(A_i\boldsymbol{\xi}_i^{s+1}+B_i\mathbf{y}_i^s-\mathbf{b}_i)$.
11: Robot i sends/receives updates to/from neighbors.
12: **end for**
13: **for** $i=1,\dots,V$ each robot computes in parallel **do**
14: $\mathbf{y}_i^{s+1}\leftarrow$ according to (15).
15: Robot i sends/receives updates from neighbors.
16: **end for**
17: **end for**
18: **for** $i=1,\dots,V$ each robot computes in parallel **do**
19: Select $\mathbf{u}_i(1)$ and implement it.
20: Update $\boldsymbol{\xi}_i^0$.
21: **end for**
22: **end for**

[4]The values of $\Delta\mathbf{p}_{i,j}^h$ are shared among the robots, but for their computation, we can use the computation units onboard of each dedicated robot.

$\Delta\mathbf{p}^h_{i,j}$, $\Delta\mathbf{p}^h_{j,i}$, η_i, and $\mathbf{p}_i$. Hence, $\mathbf{b}_i$ can be derived from (11) based on the values of $\mathbf{z}_i(t+k)$ computed at the previous problem instant, but using the updated values of $\Delta\mathbf{p}^h_{i,j}$, $\Delta\mathbf{p}^h_{i,j}$. We could proceed differently using the values of $\mathbf{p}_j$ and $\mathbf{p}^h_j$ computed by the neighboring robots. This will lead to an ADMM strategy with more than two sets of variables to update and requires each robot to wait for all the neighboring robots to update their decision variables in a sequential fashion. Our strategy allows all the robots to proceed in parallel with their local computations. Furthermore, the direct use of $\mathbf{p}_j$ and $\mathbf{p}^h_j$ means that the ADMM strategy operates directly on the collision avoidance constraints (that can be converted to equality constraints using nonconvex indicator functions in the cost). This leads to nonlinear equality constraints that might compromise the convergence of the ADMM algorithm.

Algorithm 1 requires the robots to exchange information at given synchronization points (Steps 11 and 15). At Step 11, each robot sends its own predicted position and orientation; that is, it sends to its neighbors a vector of size $3N \times 1$. At Step 15, each robot sends to its neighbors a vector of size $2hN \times 1$ according to the definition of $\Delta p^h_{i,j}$.

C. Convergence of Algorithm 1

Given the similarity of Algorithm 1 with the nonconvex ADMM proposed in [40], we can rely on the theoretical analysis of [40] to provide convergence guarantees for Algorithm 1.

Theorem 1: Let $\beta \in (0, 2)$ and $\rho > 2L/\operatorname{eig}_{\min}(B^{\mathrm{T}}B)$, $L > 0$. Assume that a feasible solution of the centralized coordination problem (9) exists. Then, Algorithm 1 converges asymptotically to a (locally) optimal solution of the coordination problem.

Proof: To prove the theorem, we show that the problem we want to solve [i.e., Problem (13)] satisfies the assumptions needed to use the convergence analysis of NADMM provided by [40]. Then, we prove the theorem using the fact that Problem (13) is equivalent (by construction) to Problem (9) when the constraints (15) are satisfied at equality.

We proceed as follows. First, we show that $J : \mathbb{R}^{n_\xi} \to \bar{\mathbb{R}}$, and matrices $A, B, \mathbf{b}$ [in (13c)] are as follows.

1) (AJ) *is a Lower Semicontinuous Function:* Note that in our problem, compared with the framework of [40], we have to consider that $\boldsymbol{\xi} \in \mathcal{F}$. We can reformulate the inner problem associated with $\boldsymbol{\xi}$ using the indicator function $\delta_{\mathcal{F}}$, which is 0 if $\boldsymbol{\xi} \in \mathcal{F}$ and ∞ otherwise. In particular, define $\mathcal{F} := \{\boldsymbol{\xi} \in \mathbb{R}^{n_\xi} | f(\boldsymbol{\xi}) \leq 0\}$, with $f(\boldsymbol{\xi})$ deriving from the definition of constraints (9b)–(9d) and (10f)–(10h). Then, the image function (AJ) will be $(AJ)(\boldsymbol{\sigma}) = \inf_{\boldsymbol{\xi}}\{J(\boldsymbol{\xi}) + \delta_{\mathcal{F}}(f(\boldsymbol{\xi}))|A\boldsymbol{\xi} = \boldsymbol{\sigma}\}$. Given that $\mathcal{F}$ is closed, $\delta_{\mathcal{F}}$ is lower semicontinuous. Consequently, the sum of lower semicontinuous functions is still lower semicontinuous [66], and the above holds.
2) $A\,\mathbf{dom}J \subseteq \mathbf{b} + \mathbf{Range}(B)$*:* This holds given that our B matrix in (12c) has full column rank.

The above shows that our problem satisfies the assumptions in [40] for the convergence of NADMM.

Algorithm 2 Asynchronous Distributed NMPC

1: **for** i = 1, ..., V **do**
2: Given $\boldsymbol{\xi}_i^0$, $\mathbf{y}_i^0$, $\boldsymbol{\lambda}_i^0$, $\rho > 0$, $\beta \in (0, 2]$, $\epsilon_i \geq 0$.
3: **end for**
4: **for** $i = 1, \ldots, V$ in parallel **do**
5: **for** $t_i = 0, 1, 2, \ldots$ **do**
6: Check current neighbors' positions.
7: **for** $s_i = 1, \ldots, \mathtt{iter}_{\max}$ **do**
8: Check current neighbors' positions.
9: **if** No updated information from the neighbors **then**
10: Use shifted neighbor's information.
11: **end if**
12: Update $\mathbf{b}_i$.
13: $\hat{\boldsymbol{\lambda}}_i^s \leftarrow \boldsymbol{\lambda}_i^s - \rho(1-\beta)(A_i\boldsymbol{\xi}_i^{s+1} + B_i\mathbf{y}_i^s - \mathbf{b}_i)$.
14: $\boldsymbol{\xi}_i^{s+1} \leftarrow \operatorname{argmin}_{\boldsymbol{\xi}_i \in \mathcal{F}_i} \mathcal{L}_i\left(\boldsymbol{\xi}_i, \mathbf{y}_i^s, \hat{\boldsymbol{\lambda}}_i^s, \epsilon_i\right)$.
15: $\boldsymbol{\lambda}_i^{s+1} \leftarrow \hat{\boldsymbol{\lambda}}_i^s + \rho(A_i\boldsymbol{\xi}_i^{s+1} + B_i\mathbf{y}_i^s - \mathbf{b}_i)$.
16: Robot i sends updates to the neighbors.
17: Check for new neighbors' information.
18: **if** No updated information from the neighbors **then**
19: Use shifted neighbor's information.
20: Inflate collision-free region by ϵ_i (16).
21: **end if**
22: $\mathbf{y}_i^{s+1} \leftarrow$ according to (15).
23: Robot i sends updates to the neighbors.
24: **end for**
25: Select $\mathbf{u}_i(1)$ and implement it.
26: Update $\boldsymbol{\xi}_i^0$.
27: **end for**
28: **end for**

Second, for ρ large enough, under the (nonrestrictive) assumption that the subproblems in Algorithm 1 admit a feasible (not necessarily unique) solution, we can establish the convergence of our algorithm. Using [41, Th. 5.6] in our framework, the following holds for the iterates generated by Algorithm 1.

1) The residual $(A\boldsymbol{\xi}^k + B\mathbf{y}^k - \mathbf{b})_{k\in\mathbb{N}}$ vanishes with $\min_{i\leq k} \|A\mathbf{x}^i + B\mathbf{y}^i - \mathbf{b}\| = O(1/(k)^{1/2})$.
2) All accumulation points $(\boldsymbol{\xi}, \mathbf{y}, \boldsymbol{\lambda})$ of $(\boldsymbol{\xi}^k, \mathbf{y}^k, \boldsymbol{\lambda}^k)_{k\in\mathbb{N}}$ satisfy the KKT conditions $0 \in \partial(\boldsymbol{\xi}) + A^{\mathrm{T}}\mathbf{y}$, $0 \in B^{\mathrm{T}}\mathbf{y}$, $A\boldsymbol{\xi} + B\mathbf{y} = \mathbf{b}$ and attain the same cost $J(\boldsymbol{\xi}, \mathbf{y})$, this being the limit sequence $(\mathcal{L}_\rho(\boldsymbol{\xi}^k, \mathbf{y}^k, \boldsymbol{\lambda}^k))_{k\in\mathbb{N}}$.

Hence, our synchronous algorithm converges asymptotically to a locally optimal solution. ■

IV. Asynchronous Distributed NMPC

Algorithm 1 proposes a strategy to solve the coordination of multiple robots in a distributed way. The main limitation of this algorithm is related to the amount of information exchanged and idle time at every iteration of the solver. Step 9 of Algorithm 1 requires the solution of a constrained nonlinear optimization problem. Its solution might take a different amount of time for each robot (depending on the number of neighbors) for every iteration of the solver. In addition, in a real setting, it might be problematic to ensure that the robots are synchronized due to packet loss. To overcome these issues, we propose a modified version of Algorithm 1 that allows asynchronous communications among the robots and mitigates the effects of packet loss. The key insight for this algorithm is to rely on the NMPC feature of providing predictions to help the robots achieve consensus.

A. Proposed Design

Algorithm 2 details our proposed asynchronous strategy. The asynchronous NADMM strategy (Steps 7–24) differs from the one used in Algorithm 1 (Steps 5–17) in the way the robots exchange information. The synchronization Steps 11 and 15 of Algorithm 1 are modified as described in the remainder of the section (Steps 8–11 and Steps 18–21 of Algorithm 2, respectively). The other steps of the algorithm are the same as in the synchronous case. Loosely speaking, the idea is the following. At every s_i iteration of the NADMM scheme, each robot is responsible to broadcast its updated information to its neighbors, but each robot does not have to wait for its neighbors to send data (we highlight this using a subscript i in the NADMM iteration counter). Every time the robot reaches a synchronization point (Steps 6, 8, and 17), it checks whether new information from the neighbors is available. If not, the robot speculates that a packet might be lost and initialize the *mitigation procedure*. The mitigation procedure consists of two main steps, that are, the choice of a safety parameter ϵ_i and the update of the neighbors' trajectories.

1) Selection of the Safety Parameter ϵ: Each robot inflates the collision region by a quantity ϵ_i proportional to the possible error robot-i's neighbors might have on the current position of robot i to preserve safety. Specifically, ϵ is defined as follows (we neglect the subscript i to indicate the robot):

$$\epsilon = n_s(t_{\text{OPT}} v_x) \tag{16}$$

where t_{OPT} is the time the robot spent to solve its local subproblem, v_x is the local longitudinal speed of the robot, and n_s is the total number of missed synchronization points in one NADMM iteration.

2) Update of the Neighbors Trajectories: Robot i continues its local NADMM iterates using the latest available predictions of the neighbors. These predictions are shifted by the required amount of steps to compensate for the asynchronous communications/packet losses (interpolating the end of the horizon based on their predicted speed).

B. Convergence of Algorithm 2

In the following, we will show that despite the lack of synchronous communication, Algorithm 2 converges to a (suboptimal) local solution of Problem (9).

Theorem 2: Under the same assumptions of Theorem 1, assuming no plant-prediction model mismatch, and under the following boundedness assumptions:

1) $t_{\text{OPT}} < t_s N$ (i.e., the solving time of the local optimization problems is smaller that the length of the horizon);
2) $n_s < N$ (i.e., the number of consecutive packet losses is smaller that the number of NMPC stages).

Algorithm 2 converges to a (locally) suboptimal solution of Problem 9.

Proof: Trivially, if no packet loss or synchronization delay occurs, Algorithm 2 converges to the same solution of Algorithm 1.

Suppose that we have two robots, namely, Robots i and j. The asynchronous behavior is caused by different local solving times and packet losses. Under the assumptions of the theorem, the local solving times are bounded, as well as the packet losses. In addition, the open-loop predictions will match the closed-loop ones.

Suppose that at time instant t, Robot i is faster than robot j to solve its local problem, that is, $t^i_{\text{OPT}} < t^j_{\text{OPT}}$. When robot i reaches the coordination checkpoint with robot j, it will not find any updated information from Robot j (note that, from the perspective of Robot i, it does not matter whether there was a packet loss or a smaller computation time). To proceed, Robot i accounts for the additional distance it will cover due to the smaller computation time by enlarging its collision avoidance region by ϵ_i according to (16). In addition, using the latest information from Robot j (i.e., the information received at time $t-1$) shifted in time, Robot i optimizes by *speculating* on the behavior of robot j. Using robot-j predictions, Robot i is virtually synchronizing with Robot j on a more conservative problem formulation that accounts for different computation times and delayed information (due to the packet loss). Similarly, when Robot j will reach the synchronization point with Robot i, it will find the updated information from Robot i (which already moved forward in the optimization process). Given that robot i is not aware of t^j_{OPT}, robot j enlarges its feasible region by ϵ_j to anticipate possible collisions due to the mismatch. In addition, given that Robot i already moved forward, Robot j will anticipate for its local delay using the shifted prediction of Robot i. Under the boundedness assumptions of the theorem, because of the use of the NMPC predictions to virtually synchronize the robots, Algorithm 2 can be casted in a synchronous one. Compared with the synchronous algorithm, however, Algorithm 2 solves the modified local-NMPC problems (with a more conservative feasible region), whose level of suboptimality is proportional to the computation time of the local subproblem and to the number of packet losses. ■

C. Discussion

The proposed asynchronous strategy is tailored to the NMPC framework. Without the use of the predictions to virtually synchronize the robots, it would not be possible to provide any guarantee related to the convergence of the algorithm. In addition, we will show that the boundedness assumptions required by Algorithm 2 are reasonable based on the simulation results provided in Section V.

The strongest assumption we made is related to the no-model-mismatch assumption. We will show in the simulation and experimental results how the proposed algorithm performs if this assumption does not hold (e.g., Fig. 6(c) or results presented in Section V-D). In general, the plant-prediction model mismatches could be compensated by adding more complexity to the NMPC prediction model to improve the quality of the predictions and using a more conservative choice of the safety parameter based on the expected deviation of the plant by the predicted trajectories. In addition, recall that the proposed strategy operates at planning level. A low-level controller can help to mitigate the effect of the plant-prediction model mismatch.

Compared with an approach that only computes local paths based on the previously computed neighbors' predictions, our

asynchronous approach still allows the individual robots to reach consensus based on the neighbors expected behavior. In addition, by allowing multiple exchanges of information within a sampling Instant, our algorithm is more resilient to packet losses that can occur when exchanging a single individual trajectory. That is, suppose that Robot i reaches the first coordination checkpoint before the other robots. With no ADMM exchanges, Robot i will miss updates from the neighbors that might arrive just a few millisecond later. Because of the multiple ADMM iterates, within the same sampling instant, Robot i can go back to the checkpoint and check again for available information.

The asynchronous approach allows the individual robots to mitigate packet losses and to use non-blocking information exchanges. Compared with the synchronous implementation, however, this approach leads to more conservative behaviors due to the preventive enlargement of the collision-free region of the individual robots by ϵ. The proposed mitigation strategy is currently based on the local information the robot has and by relying on communication. This is to show and discuss the behavior of the proposed algorithm in the worst-case scenario of minimal sensor information. Alternatively, we could combine the information received from the other robots with local onboard sensor information (e.g., camera and lidar) to refine our estimate and further improve the resilience to communication faults. Otherwise, an additional detection module or a model of the packet loss for the selected communication network might help reduce the conservatism in the solution. Finally, the local solvers will always introduce a delay due to different t_{OPT} values. Using a solver with real-time computation guarantees (e.g., for convex distributed MPC formulation, there are variations of gradient descent that can provide these guarantees [46], [67]), we could better anticipate this delays.

V. Numerical Results

This section shows the performance of our algorithms to control a team of autonomous vehicles, both in simulation and with real-life experiments.

To describe the vehicle dynamics in the local MPC formulations, we used the following kinematic bicycle model [68]:

$$\dot{x} = v_x \cos(\eta), \quad \dot{y} = v_x \sin(\eta), \ \dot{\eta} = \frac{v_x \tan(\omega)}{L} \tag{17a}$$

$$\dot{v}_x = a_x, \quad \dot{a}_y = (2a_x\omega + v_x\delta)\frac{v_x}{L}, \ \dot{\omega} = \delta \tag{17b}$$

where the states are the position of the vehicle $\mathbf{p} := [x\ y]^{\mathrm{T}}$, the orientation η, the velocity in the longitudinal direction v_x, the lateral acceleration a_y, and the angular velocity ω. The commands are the longitudinal acceleration a_x and the steering change rate δ, respectively. L is the length of the vehicle. Based on these quantities, we derived a_m, a_M, and r.

The commands and the states are subject to the following constraints: $a_x \in [\underline{a}_x, \bar{a}_x]$, $\delta \in [\underline{\delta}, \bar{\delta}]$, $v_x \in [\underline{v}_x, \bar{v}_x]$, $a_y \in [\underline{a}_y, \bar{a}_y]$, $\eta \in [\underline{\eta}, \bar{\eta}]$, and $\omega \in [\underline{\omega}, \bar{\omega}]$. The values of bounds are reported in Table I for both the simulation environment and the experimental platform. In the reminder of the section, we compare the performance of Algorithms 1 and 2 with the one of the centralized approach. For the comparison, we made the following assumptions.

1) The communication radius is sufficiently large that each vehicle is aware of the position of the neighbors from the beginning of the experiment.
2) There is no model mismatch between the MPC prediction model and the plant model; that is, we assume that both models are discretized at 0.1 s, and the planned input is applied to the plant instantaneously (only for the simulations).

These assumptions are needed to have a fair comparison with the centralized approach, whose computation time will be too large for a more realistic implementation. Later, in the section, we will relax these assumptions (also in simulation) for the asynchronous design to show how it performs in the presence of model mismatch and smaller communication radius.

We tuned the centralized algorithm to ensure that the coordination has a feasible solution (to satisfy the assumptions of the theorems). Then, we kept the same tuning for both the synchronous and asynchronous algorithms (see Table I). In addition, we set the ADMM iterations of the synchronous algorithm to ten to show how it can reach the accuracy of the centralized one with a sufficiently large number of iterations (recall that the ADMM ensure convergence only asymptotically), and we set the ADMM iterations of the asynchronous one to two to work with performance close to real time. We could reduce the number of iterations of the synchronous algorithm by enlarging its feasible region to account for the early termination of the ADMM solver. However, we want to show first how the synchronous algorithm solves the original problem in practice. Then, later, to evaluate the computation time of the synchronous algorithm, we set the number of iterations to two to compare with the asynchronous one, and we enlarged the feasible region by $t_s v^{\text{ref}}$ m to compensate for the early termination of the solver.

We relied on FORCES Pro [51] to solve the local nonconvex optimization problems. For the simulations, the algorithms are implemented in MATLAB R2017b running on a Windows OS

TABLE I

Constraint Bounds and Relevant Quantities Used for the Simulations and for the Experiments

Quantity	Unit	Simulation	Experiment
$\underline{a}_x$	m/s^2	-2	-2
$\bar{a}_x$	m/s^2	6	6
$\bar{\delta} = -\underline{\delta}$	rad/s	0.5	0.5
$\bar{a}_y = -\underline{a}_y$	m/s^2	3	3
$\bar{\eta} = -\underline{\eta}$	deg	30	30
$\underline{v}_x$	m/s	0.1	0
$\bar{v}_x$	m/s	15	3
L	m	4	0.4
v_{ref}	m/s	10	0.75
N	samples	20	30
ρ	-	0.001	0.01

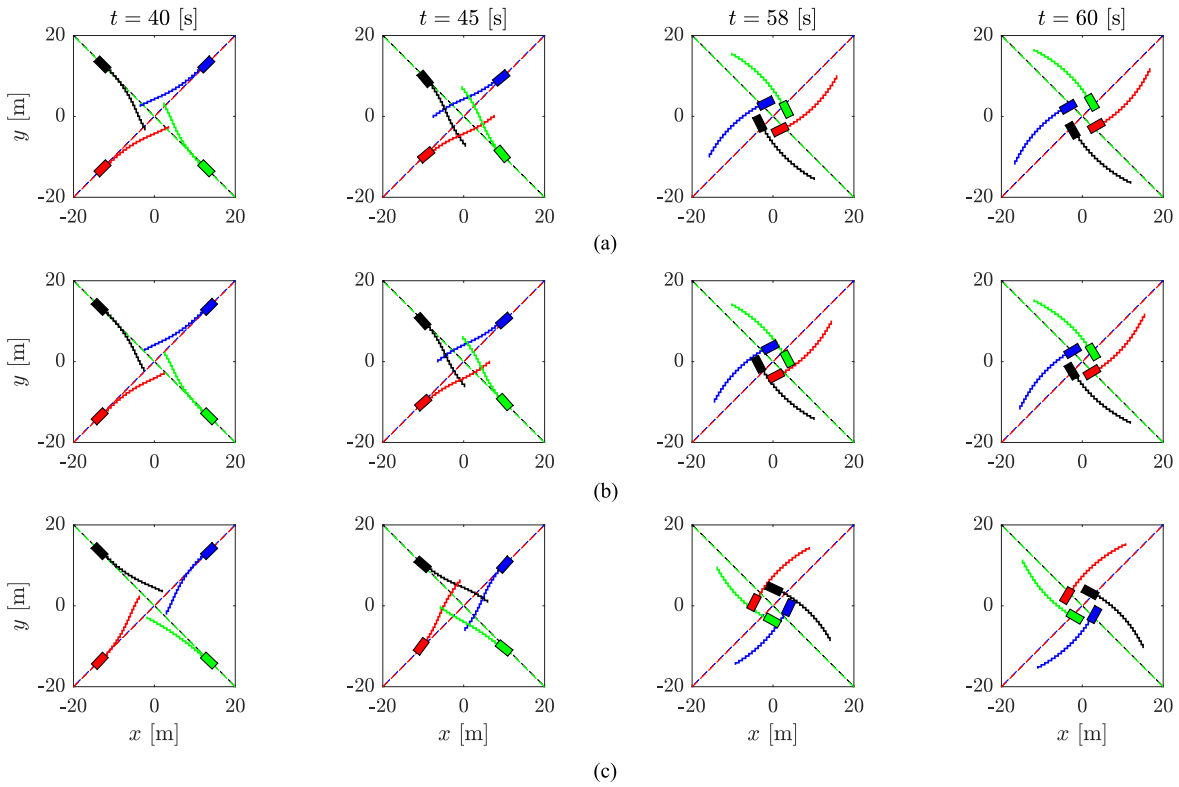


Fig. 4. Performance of Algorithms 1 and 2 (third row) compared with the centralized formulation in an intersection crossing scenario (Scenario 1). The figure compares the behavior of the three methods at critical time instances of the coordination. (a) Centralized algorithm (baseline). (b) Synchronous algorithm (see Algorithm 1). (c) Asynchronous algorithm (see Algorithm 2).

with an Intel[5] Xeon[5] CPU @ 3.60 GHz. Each vehicle runs on an independent MATLAB worker and communicates with the other vehicles using the message passing algorithm supported by MATLAB. For the synchronous algorithm, we enforced the synchronization with barrier functions. For the asynchronous implementation, we allowed each worker to broadcast its information with a time stamp to have non-blocking interactions. Both the unpredictable optimization times and the communication strategy will introduce random packet losses/asynchronicity. We do not explicitly model packet loss. Each time a robot reaches a synchronization point, if no information is available, we assume that the information is lost, and the robot moves forward according to the strategy proposed in Algorithm 2. In the remainder of the section, we will depict the vehicles as colored rectangles. In addition, the global path that the vehicles are following will be represented in dashed lines, whose color matches the ones of the associated vehicle. Furthermore, we will represent the trajectory that each vehicle will follow along the horizon at time t with solid lines, whose color matches the one of the associated vehicles. A video accompanies the paper [69].

A. Uncontrolled Intersection Crossing With Four Vehicles

1) Comparison Without Model Mismatch: Fig. 4 compares the centralized design with the distributed approaches. All the three approaches are able to cross the intersection.

[5]Registered trademark.

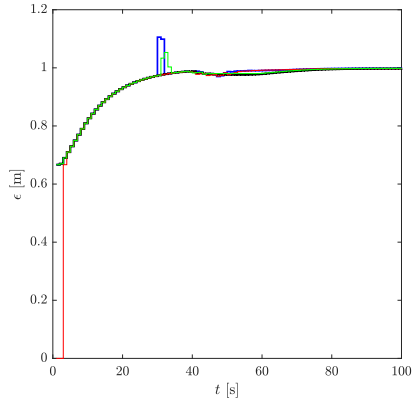


Fig. 5. Safety parameter ϵ_i $(i = 1, \ldots, 4)$ in Scenario 1. The colors match the ones of the vehicles in Fig. 4.

The asynchronous implementation, however, converges to a slightly different solution compared with the synchronous and centralized ones. This is caused by the limited number of iterations we used for the ADMM strategy of the asynchronous algorithm to reduce the computational overhead. In addition, the synchronous algorithm is able to recover the solution of the centralized approach despite the distributed implementation. Both the synchronous (second row of Fig. 4) and asynchronous (bottom row of Fig. 4) approaches are able to preserve safety (no collision during the coordination) because of the local hard collision avoidance constraints and the regularization term in

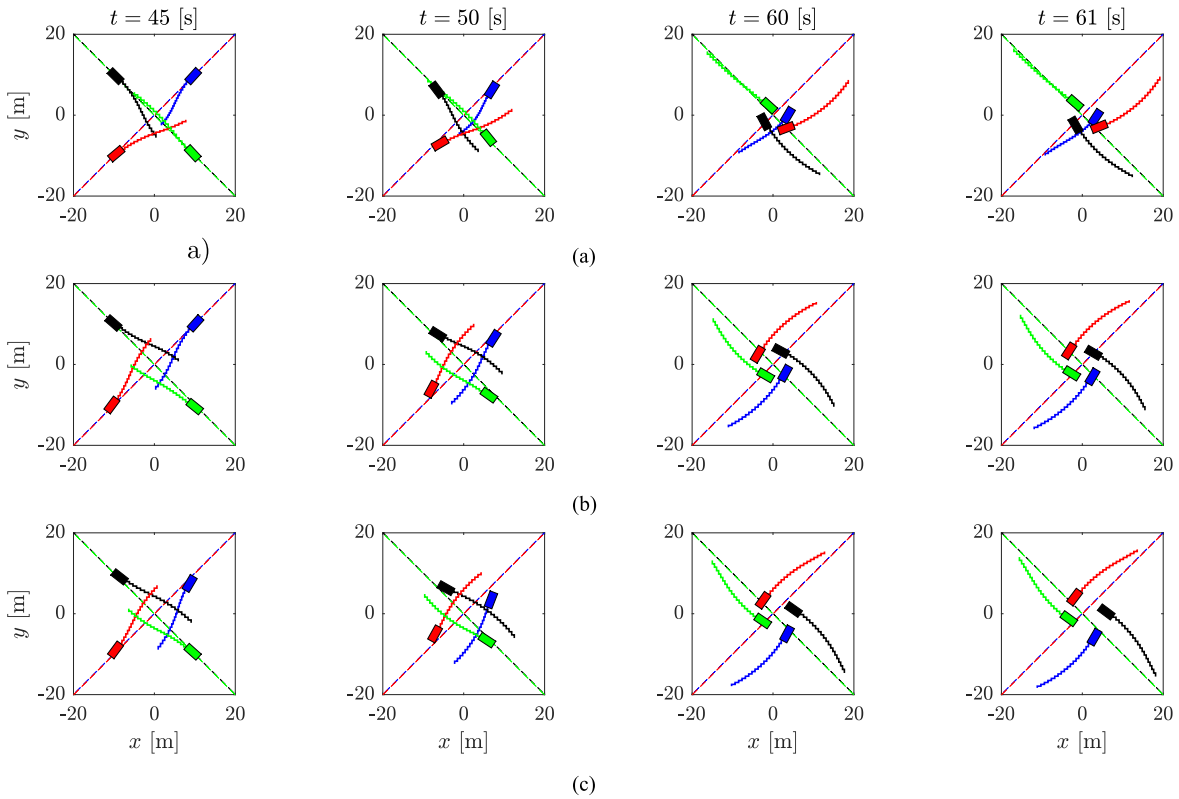


Fig. 6. Performance of Algorithm 2 (a) without model mismatch and without safety parameter $\epsilon_i = 0$, (b) without model mismatch, and (c) with model mismatch and limited communication range (Scenario 1).

the augmented Lagrangian that enforces consistency between the local subproblems. For the asynchronous implementation, we run multiple (≈100) instances of the proposed scenario to test the robustness of the approach to asynchronous communications and packet losses. Because of the adaptive selection of the safety parameter ϵ, the algorithm preserved safety in all the experiments.

Fig. 5 shows the changes in the safety parameter ϵ during the simulation depicted in Fig. 4. Note that around 30 s, the ϵ values of the blue and green vehicles have a small increase. This is because the computation time of the local solutions sharply increases due to the increase in the number of active constraints along the prediction horizon (the robots are approaching the intersection, and the coordination should be much tighter). Afterward, the values slightly decrease, because the robots decrease their speed to cross the intersection, reducing the predicted distance they plan to cover between two time steps.

2) Comparison With and Without Safety Parameter ϵ_i: Fig. 6(a) and (b) shows the importance of the safety parameter ϵ required by the asynchronous algorithm according to Theorem 2. As Fig. 6(a) shows, without ϵ, due to the presence of delays and packet losses, the algorithm is not able to guarantee collision-free trajectories (as the third and fourth columns show).

3) Comparison With Model Mismatch: Fig. 6(c) shows the performance of Algorithm 2 in a more realistic setting. In particular, we consider a model mismatch between the plant and the prediction model; that is, the plant is discretized

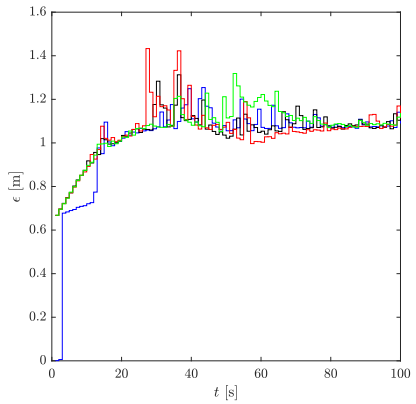


Fig. 7. Safety parameter ϵ_i in Scenario 1 with model mismatch.

at 0.01 s, while the prediction model is discretized at 0.1 s. This means that while the planner is computing a feasible trajectory, the vehicles keeps using the previously computed commands to compensate for the lack of new information. In addition, we considered that the vehicles can only communicate within a communication radius of 100 m (we selected this range according to a recent study on vehicular ad hoc network (VANET) communication technologies [70]). Note that despite the model mismatch and the limited communication range, Algorithm 2 is still able to find a suitable crossing strategy. Note that the safety parameter ϵ also adapts compared with the behavior of ϵ without model mismatch (as Figs. 5 and 7 depict).

TABLE II

TABLE COMPARED THE PERFORMANCE IN TERMS OF OVERALL TIME (SYNCHRONIZATION PLUS ONLINE OPTIMIZATION TIMES) OF THE PROPOSED DISTRIBUTED ALGORITHMS WITH THE CENTRALIZED ONE IN THE INTERSECTION CROSSING SCENARIO (SCENARIO 1). THE TIMES ARE CALCULATED FOR EACH VEHICLE AND OVER 11 SIMULATIONS

	Centralized Design	Synchronous Design (Algorithm 1)				Asynchronous Design (Algorithm 2)			
Max computation time (sec)	1.2168	1.9810	1.6663	1.6663	1.7756	0.1276	0.0778	0.0948	0.1235
90 percentile of the computation time (sec)	0.3277	0.5931	0.5946	0.5946	0.5792	0.0427	0.0337	0.0337	0.0391
Mean of the computation time (sec)	0.2283	0.2164	0.2150	0.2150	0.2152	0.0315	0.0276	0.0272	0.0296

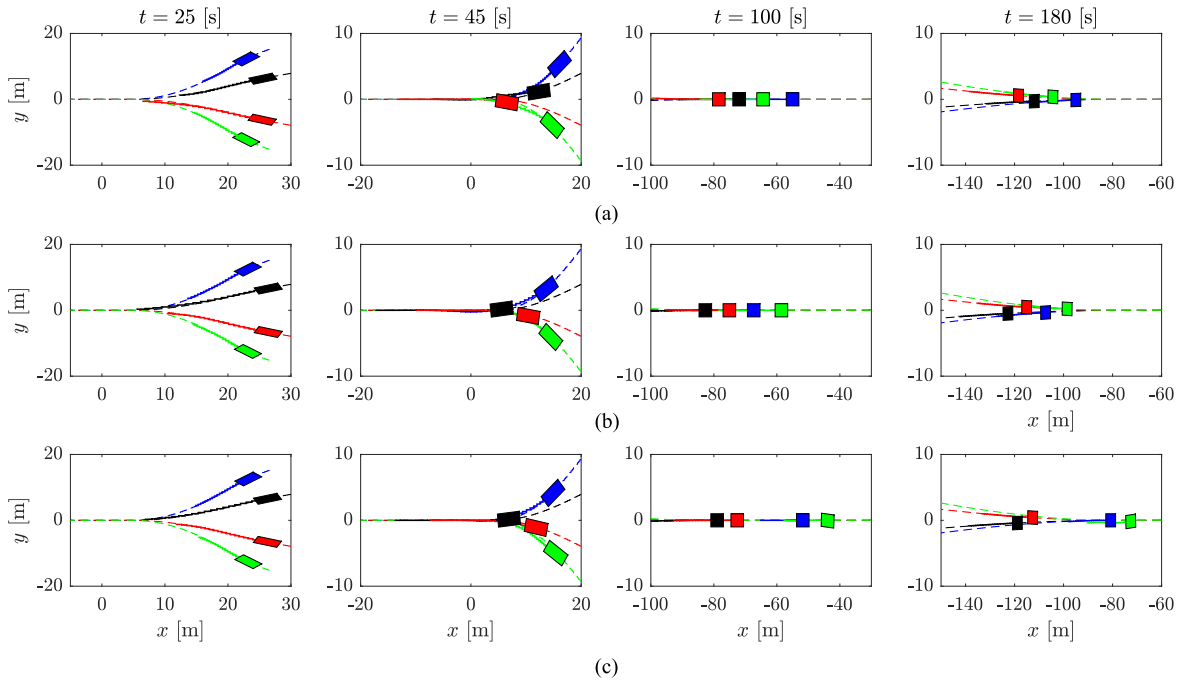


Fig. 8. Performance of Algorithms 1 and 2 compared with the centralized formulation in Scenario 2. Due to significant packet losses and delays, the asynchronous algorithm becomes more conservative compared with the synchronous and centralized ones. (a) Centralized algorithm (baseline). (b) Synchronous algorithm (see Algorithm 1). (c) Asynchronous algorithm (see Algorithm 2).

4) Coordination Time: Table II provides a comparison of the coordination time of the three algorithms over 11 runs of Scenario 1. Note that the proposed asynchronous strategy outperforms the synchronous and the centralized approach in terms of computation time. The 90 percentile metrics shows that the asynchronous planner returns a solution within 0.05 s, almost an order of magnitude smaller than the synchronous and the centralized methods that need 0.6 and 0.3 s, respectively. Note that the large coordination times of the synchronous algorithm are mostly due to the synchronization overhead. If no synchronization overhead is present, the synchronous algorithm will perform similar to the asynchronous one in terms of coordination times.

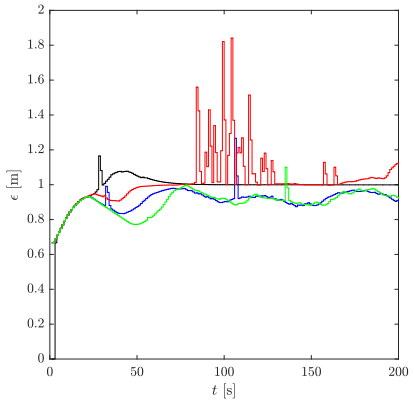


Fig. 9. Values of ϵ_i $(i = 1, \ldots, 4)$ used to inflate the collision-free region to protect the vehicles from collisions due to packet losses in Scenario 2.

B. Platooning With Four Vehicles

1) No Model Mismatch: Fig. 8 compares the centralized design with the distributed approaches in the platoon formation scenario, that is, Scenario 2.[6] We compared the algorithms with no model mismatch and with an infinite communication radius. We selected this experiment to show how the asynchronous algorithm can some time be over conservative if a large number of packet losses occur or the vehicles go heavily

[6]We only report the simulation results with no model mismatch for this scenario, given that the results do not add additional information compared with the previous ones.

TABLE III

TABLE COMPARED THE PERFORMANCE IN TERMS OF OVERALL TIME (SYNCHRONIZATION PLUS ONLINE OPTIMIZATION TIMES) OF THE PROPOSED DISTRIBUTED ALGORITHMS WITH THE CENTRALIZED ONE IN THE PLATOON FORMATION SCENARIO (SCENARIO 2). THE TIMES ARE CALCULATED FOR EACH VEHICLE AND OVER 11 SIMULATIONS

	Centralized Design	Synchronous Design (Algorithm 1)				Asynchronous Design (Algorithm 2)			
Max computation time (sec)	0.6639	1.9769	1.9895	1.9895	1.9888	0.1209	0.080	0.1298	0.1015
90 percentile of the computation time (sec)	0.3278	0.3891	0.3451	0.3451	0.3476	0.0388	0.0322	0.0365	0.0360
Mean of the computation time (sec)	0.2350	0.1612	0.1572	0.1572	0.1501	0.0233	0.0247	0.0254	0.0229

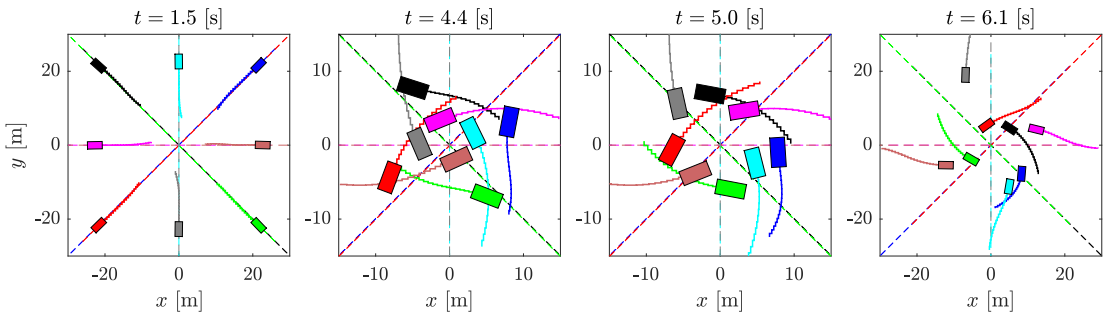


Fig. 10. Performance of Algorithm 2 in an intersection crossing scenario with eight vehicles involved.

TABLE IV

TABLE REPORTS THE OPTIMIZATION TIMES OF THE PROPOSED DISTRIBUTED COORDINATION STRATEGY (USING ALGORITHM 2) IN AN INTERSECTION CROSSING SCENARIO WITH EIGHT VEHICLES. THE TIMES ARE CALCULATED FOR EACH VEHICLE AND OVER 11 SIMULATIONS

	Asynchronous Design (Algorithm 2)							
Max computation time (sec)	0.6757	0.3783	0.3783	0.4494	0.4918	0.3657	0.5146	0.5173
90 percentile of the computation time (sec)	0.1253	0.1213	0.1213	0.1162	0.1234	0.1087	0.1119	0.1209
Mean of the computation time (sec)	0.1695	0.1523	0.1523	0.1468	0.1497	0.1485	0.1418	0.1485

out of sync (we lowered the worker process priority to worsen the performance of the algorithm). Note that for this scenario, the synchronous algorithm shows again similar performance to the centralized one. The distributed algorithms are able to coordinate the vehicles to merge into a platoon formation and later split to proceed on different lanes. Concerning the asynchronous algorithm, the obtained solution is more conservative than the synchronous one. The conservatism can be explained by looking at the safety parameter of the red vehicle (which is in front of the green and the blue ones) depicted in Fig. 9. The spikes are associated with the substantial delay in the computation time of the red vehicle that increases its safety region to protect the neighboring vehicles from collision.

2) Coordination Time: Table III provides a comparison of the coordination time of the three algorithms over 11 runs of Scenario 2. Similar to the previous scenario, the proposed asynchronous strategy outperforms the synchronous and the centralized approach in terms of computation time. The 90 percentile metrics shows that the asynchronous planner returns a solution within 0.04 s, an order of magnitude smaller than the synchronous and the centralized methods that need 0.4 and 0.3 s, respectively.

C. Intersection Crossing With Eight Vehicles

Fig. 10 shows a scenario in which eight vehicles coordinate at an intersection crossing scenario. As the figure shows, the vehicles are able to successfully cross the intersection without collisions by exchanging information according to Algorithm 2. The tuning of the algorithm (e.g., the values of N and ρ) is the same compared with the one used in the previous simulations. However, due to the increasing number of neighbors for each individual robot, the computation time of the method increases (as Table IV shows), and it is approximately three times slower than the same crossing scenario with just four vehicles. Notice, however, that the computation time is still close to the sampling time of the system according to the mean and 90 percentile results. In addition, we expect further speedup when running each of the vehicles on independent machines (the computer used for the experiments has only four independent cores).

D. Experimental Results

We tested Algorithm 2 using a team of three small-scale autonomous vehicles. The vehicles are based on the Waveshare JetRacer Pro AI[5] kit, a platform suited to replicate the behavior of full scale vehicles. The vehicles use RC380 carbon brushed motors for locomotion. The steering is controlled by an E6001 servo motor. Wi-Fi connectivity is provided by a 2.4/5-GHz Dual-band Wireless AC8265 module. An electronic speed controller (ESC) is used to interface the Jetson Nano board with the driving motor and steering servomotor. For our experiments, we used a Motive OptiTrack[5] external motion capture

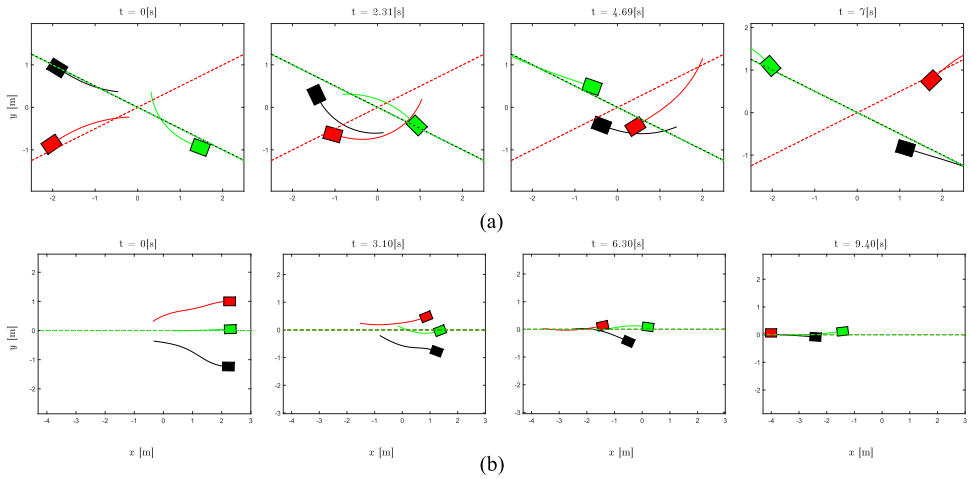


Fig. 11. Experimental results of both the unsupervised intersection crossing and lane merging scenarios. (a) Unsupervised intersection crossing. (b) Unsupervised lane merging.

TABLE V

TABLE PROVIDES THE COMPUTATION TIMES OF ALGORITHM 2 OBTAINED DURING THE EXPERIMENTS

Experiments using Algorithm 2	Max computation time (sec)			Mean computation time (sec)			90 percentile of computation time (sec)		
Intersection crossing	0.0547	0.0528	0.0487	0.0253	0.0250	0.0221	0.0323	0.0339	0.0275
Lane merging	0.0452	0.0314	0.0394	0.0243	0.0225	0.0242	0.0313	0.0270	0.0290

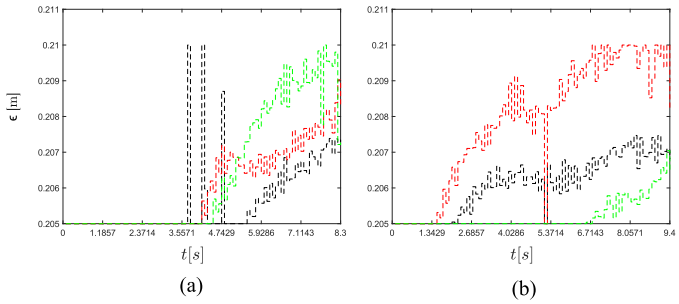


Fig. 12. Values of ϵ_i $(i = 1, \ldots, 3)$ used to inflate the collision-free region of each vehicle due to asynchronous communications during the experiments. (a) Unsupervised intersection crossing. (b) Unsupervised lane merging.

system to analyze the performance of the motion planning algorithm alone. The algorithm runs on a computational unit with a AMD Ryzen 7 5800H with Radeon Graphics 3.20 GHz and Ubuntu 20.04. The commands are sent to the robots using a dedicated ROS network.

The dimension of the map has been adapted to match the laboratory space availability and the size of the mobile robots. The weights used in the algorithm have been consequently adjusted. Fig. 11 shows the predicted trajectories (solid lines) and the position of the robots (colored boxed) measured by the motion tracking system for the following: 1) a crossing scenario and 2) a merging scenario at given time instances. The dashed lines highlight the high-level reference path. The vehicles in both scenarios are able to safely complete the maneuvers without collisions. The computation time of the algorithm is comparable to the results obtained in simulation, as Table V shows. In addition, the effects of the asynchronous communications are compensated by the activation of the safety parameter during the maneuver, as shown in Fig. 12. A full video of the experiments is provided in [69]. These experiments show the potential of the proposed scenario for real-life applications and to deal with realistic communication networks.

VI. CONCLUSION AND FUTURE WORK

We proposed two distributed local motion planners based on the NMPC for the coordination of autonomous robots. The proposed algorithms allow the robots to communicate and agree on a common safe navigation strategy without the need of a central coordinator. The two algorithms differ in the way the communication among the robots is handled. In particular, the synchronous distributed NMPC design allows for a synchronous exchange of information among the robots and ensures convergence to a (locally) optimal solution of the coordination problem. The asynchronous distributed NMPC design allows for an asynchronous exchange of information among the robots and ensures convergence to a (locally) suboptimal solution of the coordination problem.

We compared the proposed designs for the control of autonomous vehicles at a crossing and in a platoon formation scenario, both in simulation and with experiments. The asynchronous algorithm was able to safely accomplish the planning goals, while dealing with packet losses (caused by the robots going out of synch during the coordination). In addition, the asynchronous algorithm drastically reduced the local optimization times and the communication overhead by an order of magnitude, compared with a centralized implementation of the coordination strategy and the synchronous algorithm.

The proposed algorithms are general and could be applied for the distributed coordination of vessels, drones, or other types of mobile robots. This is part of our future research.

REFERENCES

[1] Bureau of Transportation Statistics, U.S. Department of Transportation. *Distribution of Transportation Fatalities by Mode*. Accessed: Jan. 2022. [Online]. Available: https://www.bts.gov/content/distribution-transportation-fatalities-mode

[2] Economic Commision for Europe. (2017). *Statistics of Road Traffic Accidents in Europe and North America*. [Online]. Available: https://www.unece.org/fileadmin/DAM/trans/main/wp6/publications/RAS-2017.pdf

[3] European Commission. (2018). *Europe on the Move: Commission Completes Its Agenda for Safe, Clean and Connected Mobility*. [Online]. Available: https://ec.europa.eu/transport/modes/road/news/2018-05-17-europe-on-the-move-3_en

[4] (2018). *Connected & Automated Mobility: For a More Competitive Europe*. [Online]. Available: https://ec.europa.eu/transport/sites/transport/files/3rd-mobility-pack/3rd-mobility-pack-factsheets-automatedconnected.pdf

[5] F. E. Schneider and D. Wildermuth, "A potential field based approach to multi robot formation navigation," in *Proc. IEEE Int. Conf. Robot., Intell. Syst. Signal Process.*, vol. 1, Oct. 2003, pp. 680–685.

[6] H. G. Tanner and A. Kumar, "Towards decentralization of multi-robot navigation functions," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 4132–4137.

[7] R. Gayle, W. Moss, M. C. Lin, and D. Manocha, "Multi-robot coordination using generalized social potential fields," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 106–113.

[8] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, Mar. 1997.

[9] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*. Berlin, Germany: Springer, 2011, pp. 3–19.

[10] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, 1998.

[11] M. Cap, P. Novak, A. Kleiner, and M. Selecky, "Prioritized planning algorithms for trajectory coordination of multiple mobile robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 835–849, Jul. 2015.

[12] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *J. Artif. Intell. Res.*, vol. 31, pp. 591–656, Mar. 2008.

[13] L. Bruni, A. Colombo, and D. Del Vecchio, "Robust multi-agent collision avoidance through scheduling," in *Proc. 52nd IEEE Conf. Decis. Control*, Florence, Italy, Dec. 2013, pp. 3944–3950.

[14] M. R. Hafner, D. Cunningham, L. Caminiti, and D. Del Vecchio, "Cooperative collision avoidance at intersections: Algorithms and experiments," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1162–1175, Sep. 2013.

[15] A. Colombo and D. Del Vecchio, "Least restrictive supervisors for intersection collision avoidance: A scheduling approach," *IEEE Trans. Autom. Control*, vol. 60, no. 6, pp. 1515–1527, Jun. 2015.

[16] H. Ahn and D. D. Vecchio, "Safety verification and control for collision avoidance at road intersections," *IEEE Trans. Autom. Control*, vol. 63, no. 3, pp. 630–642, Mar. 2018.

[17] Y. Arai, T. Fujii, H. Asama, H. Kaetsu, and I. Endo, "Collision avoidance in multi-robot systems based on multi-layered reinforcement learning," *Robot. Auto. Syst.*, vol. 29, no. 1, pp. 21–32, Oct. 1999.

[18] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," in *Innovations in Multi-Agent Systems and Applications*. Berlin, Germany: Springer, 2010, pp. 183–221.

[19] H. M. La, R. Lim, and W. Sheng, "Multirobot cooperative learning for predator avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 52–63, Jan. 2015.

[20] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 285–292.

[21] S.-H. Oh and J. Suk, "Evolutionary controller design for area search using multiple UAVs with minimum altitude maneuver," *J. Mech. Sci. Technol.*, vol. 27, no. 2, pp. 541–548, Feb. 2013.

[22] H. Zhu and J. Alonso-Mora, "Chance-constrained collision avoidance for MAVs in dynamic environments," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 776–783, Apr. 2019.

[23] L. Ferranti, R. R. Negenborn, T. Keviczky, and J. Alonso-Mora, "Coordination of multiple vessels via distributed nonlinear model predictive control," in *Proc. Eur. Control Conf. (ECC)*, Limassol, Cyprus, Jun. 2018, pp. 2523–2528.

[24] R. Firoozi, X. Zhang, and F. Borrelli, "Formation and reconfiguration of tight multi-lane platoons," 2020, *arXiv:2003.08595*.

[25] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model predictive contouring control for collision avoidance in unstructured dynamic environments," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4459–4466, Oct. 2019.

[26] G. R. de Campos, P. Falcone, R. Hult, H. Wymeersch, and J. Sjöberg, "Traffic coordination at road intersections: Autonomous decision-making algorithms using model-based heuristics," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 1, pp. 8–21, Spring 2017.

[27] M. Zanon, S. Gros, H. Wymeersch, and P. Falcone, "An asynchronous algorithm for optimal vehicle coordination at traffic intersections," in *Proc. 20th IFAC World Congr.*, Toulouse, France, 2017, pp. 12008–12014.

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

[29] R. Hult, M. Zanon, S. Gros, and P. Falcone, "Energy-optimal coordination of autonomous vehicles at intersections," in *Proc. Eur. Control Conf. (ECC)*, Limassol, Cyprus, Jun. 2018, pp. 602–607.

[30] G. R. de Campos, P. Falcone, and J. Sjöberg, "Autonomous cooperative driving: A velocity-based negotiation approach for intersection crossing," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 1456–1461, doi: 10.1109/ITSC.2013.6728435.

[31] Y. Jiang, M. Zanon, R. Hult, and B. Houska, "Distributed algorithm for optimal vehicle coordination at traffic intersections," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 11577–11582, 2017.

[32] J. Shi, Y. Zheng, Y. Jiang, M. Zanon, R. Hult, and B. Houskal, "Distributed control algorithm for vehicle coordination at traffic intersections," in *Proc. Eur. Control Conf. (ECC)*, Limassol, Cyprus, Jun. 2018, pp. 1166–1171.

[33] A. Katriniok, P. Kleibaum, and M. Joševski, "Distributed model predictive control for intersection automation using a parallelized optimization approach," *IFAC PapersOnLine*, vol. 50, no. 1, pp. 5940–5946, Jul. 2017, doi: 10.1016/j.ifacol.2017.08.1492.

[34] F. Borrelli, T. Keviczky, and G. J. Balas, "Collision-free UAV formation flight using decentralized optimization and invariant sets," in *Proc. 43rd IEEE Conf. Decis. Control*, vol. 1, Dec. 2004, pp. 1099–1104, doi: 10.1109/CDC.2004.1428839.

[35] T. Keviczky, F. Borrelli, K. Fregene, D. Godbole, and G. J. Balas, "Decentralized receding horizon control and coordination of autonomous vehicle formations," *IEEE Trans. Control Syst. Technol.*, vol. 16, no. 1, pp. 19–33, Jan. 2008, doi: 10.1109/TCST.2007.903066.

[36] H. Zheng, "Coordination of waterborne AGVs," Ph.D. dissertation, Dept. Transp. Eng. Logistics, Delft Univ. Technol., Delft, The Netherlands, 2016.

[37] L. Chen, J. J. Hopman, and R. R. Negenborn, "Distributed model predictive control for vessel train formations of cooperative multi-vessel systems," *Transp. Res. C, Emerg. Technol.*, vol. 92, pp. 101–118, Jul. 2018.

[38] H. Zheng, R. R. Negenborn, and G. Lodewijks, "Robust distributed predictive control of waterborne AGVs—A cooperative and cost-effective approach," *IEEE Trans. Cybern.*, vol. 48, no. 8, pp. 2449–2461, Aug. 2018, doi: 10.1109/TCYB.2017.2740558.

[39] F. Rey, Z. Pan, A. Hauswirth, and J. Lygeros, "Fully decentralized ADMM for coordination and collision avoidance," in *Proc. Eur. Control Conf. (ECC)*, Limassol, Cyprus, Jun. 2018, pp. 825–830.

[40] A. Themelis and P. Patrinos, "Douglas-rachford splitting and ADMM for nonconvex optimization: Tight convergence results," *SIAM J. Optim.*, vol. 30, no. 1, pp. 149–181, Jan. 2020.

[41] D. Hrovat, S. Di Cairano, H. E. Tseng, and I. V. Kolmanovsky, "The development of model predictive control in automotive industry: A survey," in *Proc. IEEE Int. Conf. Control Appl.*, Dubrovnik, Croatia, Oct. 2012, pp. 295–302, doi: 10.1109/CCA.2012.6402735.

[42] W. Schwarting, J. Alonso-Mora, L. Paull, S. Karaman, and D. Rus, "Parallel autonomy in automated vehicles: Safe motion generation with minimal intervention," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Singapore, May/Jun. 2017, pp. 1928–1935.

[43] L. Ferranti et al., "SafeVRU: A research platform for the interaction of self-driving vehicles with vulnerable road users," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 1660–1666.

[44] H. Zheng, R. R. Negenborn, and G. Lodewijks, "Cooperative distributed collision avoidance based on ADMM for waterborne AGVs," in *Proc. Int. Conf. Comput. Logistics (ICCL)*, Delft, The Netherlands, 2015, pp. 181–194.

[45] H. Zheng, R. R. Negenborn, and G. Lodewijks, "Fast ADMM for distributed model predictive control of cooperative waterborne AGVs," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1406–1413, Jul. 2017.

[46] L. Ferranti and T. Keviczky, "Operator-splitting and gradient methods for real-time predictive flight control design," *J. Guid., Control, Dyn.*, vol. 40, no. 2, pp. 265–277, 2016, doi: 10.2514/1.G000288.
[47] L. Ferranti, Y. Wan, and T. Keviczky, "Fault-tolerant reference generation for model predictive control with active diagnosis of elevator jamming faults," *Int. J. Robust Nonlinear Control*, vol. 29, no. 16, pp. 5412–5428, 2019, doi: 10.1002/rnc.4063.
[48] C. V. Rao, S. J. Wright, and J. B. Rawlings, "Application of interior-point methods to model predictive control," *J. Optim. Theory Appl.*, vol. 99, no. 3, pp. 723–757, Dec. 1998.
[49] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009, doi: 10.1137/080716542.
[50] Y. Nesterov, "Smooth minimization of non-smooth functions," *Math. Program.*, vol. 103, no. 1, pp. 127–152, 2005.
[51] A. Domahidi and J. Jerez. (Jul. 2014). *FORCES Professional*. Embotech GmbH. [Online]. Available: http://embotech.com/FORCES-Pro
[52] G. Torrisi, D. Frick, T. Robbiani, S. Grammatico, R. S. Smith, and M. Morari. (2017). *FalcOpt: First-order Algorithm via Linearization of Constraints for OPTimization*. [Online]. Available: https://github.com/torrisig/FalcOpt
[53] E. F. Camacho, T. Alamo, and D. M. de la Peña, "Fault-tolerant model predictive control," in *Proc. IEEE 15th Conf. Emerg. Technol. Factory Automat. (ETFA)*, Sep. 2010, pp. 1–8.
[54] H. A. Izadi, Y. Zhang, and B. W. Gordon, "Fault tolerant model predictive control of quad-rotor helicopters with actuator fault estimation," *IFAC Proc. Volumes*, vol. 18, no. 1, pp. 6343–6348, 2011.
[55] X. Yang and J. M. Maciejowski, "Fault-tolerant model predictive control of a wind turbine benchmark," *IFAC Proc. Volumes*, vol. 45, no. 20, pp. 337–342, Jan. 2012.
[56] H. A. Izadi, B. W. Gordon, and Y. Zhang, "Decentralized model predictive control for cooperative multiple vehicles subject to communication loss," *Int. J. Aerosp. Eng.*, vol. 2011, pp. 1–13, 2011.
[57] N. T. Hung, A. M. Pascoal, and T. A. Johansen, "Cooperative path following of constrained autonomous vehicles with model predictive control and event-triggered communications," *Int. J. Robust Nonlinear Control*, vol. 30, no. 7, pp. 2644–2670, 2020, doi: 10.1002/rnc.4896.
[58] D. M. de la Peña and P. D. Christofides, "Lyapunov-based model predictive control of nonlinear systems subject to data losses," *IEEE Trans. Autom. Control*, vol. 53, no. 9, pp. 2076–2089, Oct. 2008.
[59] N. Bastianello, R. Carli, L. Schenato, and M. Todescato, "Asynchronous distributed optimization over lossy networks via relaxed ADMM: Stability and linear convergence," 2019, *arXiv:1901.09252*.
[60] E. Wei and A. Ozdaglar, "On the O(1=k) convergence of asynchronous distributed alternating direction method of multipliers," in *Proc. IEEE Global Conf. Signal Inf. Process.*, Dec. 2013, pp. 551–554.
[61] T. Faulwasser, B. Kern, and R. Findeisen, "Model predictive path-following for constrained nonlinear systems," in *Proc. 48h IEEE Conf. Decis. Control (CDC) Held Jointly 28th Chin. Control Conf.*, Dec. 2009, pp. 8642–8647, doi: 10.1109/CDC.2009.5399744.
[62] D. Lam, C. Manzie, and M. C. Good, "Model predictive contouring control for biaxial systems," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 2, pp. 552–559, Mar. 2013.
[63] P. Patrinos, P. Sopasakis, H. Sarmiveis, and A. Bemporad, "Stochastic model predictive control for constrained discrete-time Markovian switching systems," *Automatica*, vol. 50, no. 10, pp. 2504–2514, Oct. 2014.
[64] J. M. Maciejowski, *Predictive Control: With Constraints*. London, U.K.: Pearson, 2002.
[65] J. Wang and L. Zhao, "Nonconvex generalization of alternating direction method of multipliers for nonlinear equality constrained problems," 2017, *arXiv:1705.03412*.
[66] G. Wanka, *Convex Analysis*. Chemnitz, Germany: Chemnitz Univ. Technology, 2009.
[67] I. Necoara, L. Ferranti, and T. Keviczky, "An adaptive constraint tightening approach to linear model predictive control based on approximation algorithms for optimization," *Optim. Control Appl. Methods*, vol. 36, no. 5, pp. 648–666, Sep. 2015.
[68] J. A. Matute, M. Marcano, S. Diaz, and J. Perez, "Experimental validation of a kinematic bicycle model predictive control with lateral acceleration consideration," *IFAC-PapersOnLine*, vol. 52, no. 8, pp. 289–294, 2019.
[69] L. Ferranti, L. Lyons, R. Negenborn, T. Keviczky, and J. Alonso-Mora. (Mar. 2022). *Video Associated With the Submission*. [Online]. Available: https://youtu.be/o2W2OhNf5yc
[70] Y. A. D. Djilali, Y. Bakhtil, B. Kouninef, and B. Senouci, "Performances evaluation study of VANET communication technologies for smart and autonomous vehicles," in *Proc. 10th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2018, pp. 79–84.

Laura Ferranti received the Ph.D. degree from the Delft University of Technology, Delft, The Netherlands, in 2017.

She is currently an Assistant Professor with the Cognitive Robotics (CoR) Department, Delft University of Technology. Her research interests include optimization and optimal control, model predictive control, reinforcement learning, embedded optimization-based control with application in flight control, maritime transportation, robotics, and automotive.

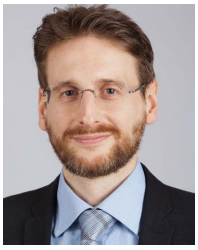
Dr. Ferranti was a recipient of the NWO Veni Grant from the Netherlands Organization for Scientific Research in 2020 and the Best Paper Award on Multi-Robot Systems at International Conference on Robotics and Automation (ICRA) 2019.

Lorenzo Lyons received the M.Sc. degree in mechanical engineering from the Polytechnic University of Milan, Milan, Italy, in 2021. He is currently pursuing the Ph.D. degree with the Cognitive Robotics (CoR) Department, Delft University of Technology, Delft, The Netherlands.

His research interests include numerical optimization, model predictive control, multi-robot motion planning applied to the automotive, and robotics.

Rudy R. Negenborn received the Ph.D. degree in distributed control from the Delft University of Technology, Delft, The Netherlands, in 2007.

He is currently a Full Professor of multimachine operations and logistics with the Transport Engineering and Logistics Section, Department of Maritime and Transport Technology, Delft University of Technology. His current research interests include multi-agent systems, distributed control, model predictive control, simulation of large-scale transport systems, and applications in (waterborne) networked transport systems.



Tamás Keviczky (Senior Member, IEEE) received the Ph.D. degree from the Control Science and Dynamical Systems Center, University of Minnesota, Minneapolis, MN, USA, in 2005.

He is currently a Full Professor with the Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands. His research interests include distributed optimization and optimal control, model predictive control, embedded optimization-based control and estimation of large-scale systems with applications in aerospace, automotive, mobile robotics, industrial processes, and infrastructure systems.

Dr. Keviczky was a co-recipient of the American Association for Clinical Chemistry (AACC) O. Hugo Schuck Best Paper Award for Practice in 2005. He is an Associate Editor of IEEE TRANSACTIONS ON AUTOMATIC CONTROL and served as an Associate Editor for *Automatica* from 2011 to 2017.

Javier Alonso-Mora (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in robotics from ETH Zürich, Zürich, Switzerland, in 2010 and 2014, respectively.

He is currently an Associate Professor with the Delft University of Technology, Delft, The Netherlands. Until October 2016, he was a Post-Doctoral Associate with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. He was also a member of the Disney Research, Zürich. His main research interests include autonomous navigation of mobile robots, with a special emphasis in multi-robot systems and robots that interact with other robots and humans. Toward the smart cities of the future, he applies these techniques in various fields, including self-driving cars, automated factories, aerial vehicles, and intelligent transportation systems.

Dr. Alonso-Mora was a recipient of the European Research Council (ERC) Starting Grant in 2021, the IEEE International Conference on Robotics and Automation (ICRA) Best Paper Award on Multi-Robot Systems in 2019 and the Veni Grant from the Netherlands Organization for Scientific Research in 2017.